January 1980 $2.00

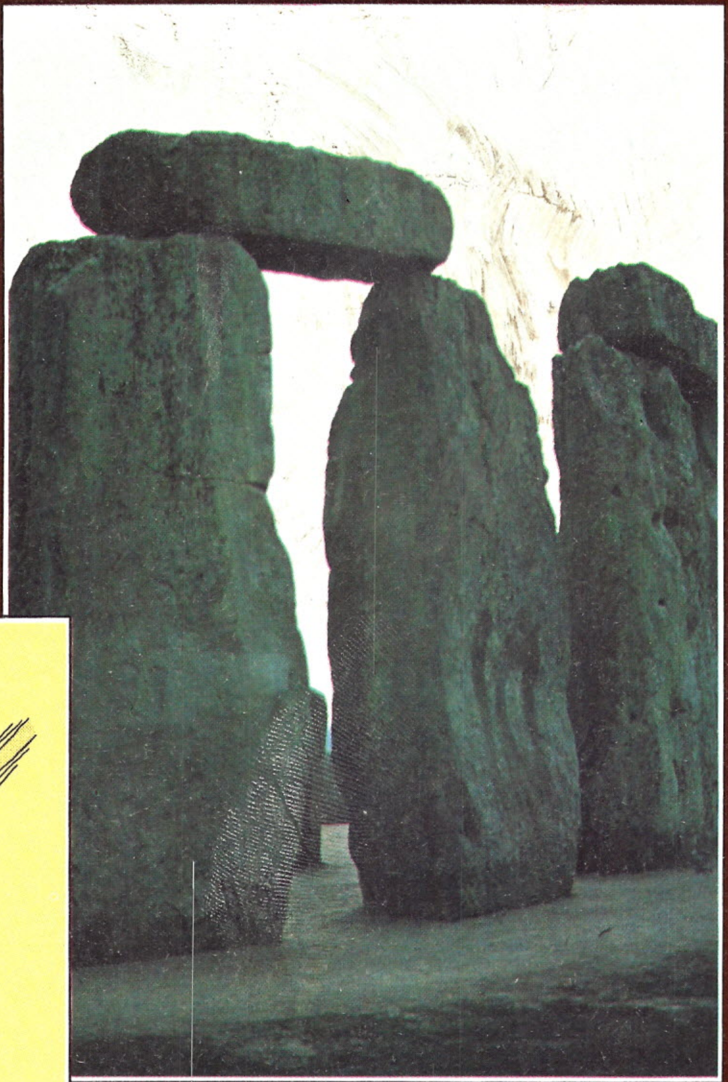
TRS-80 Line Renumbering • Wine Tasting • How to Build Programs

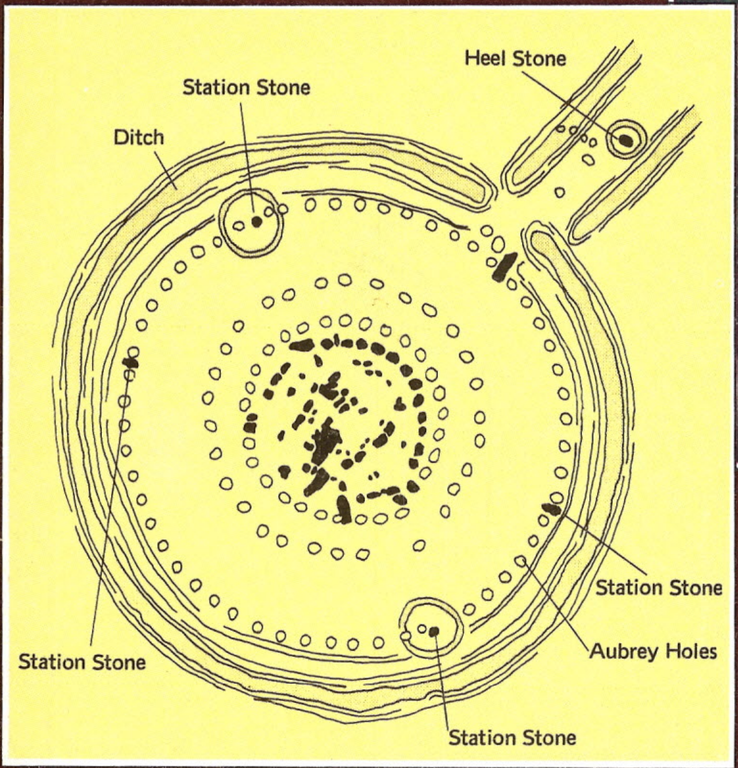
Personal Computing

For Your Home and Business

STONEHENGE

Simulating an Ancient Computer



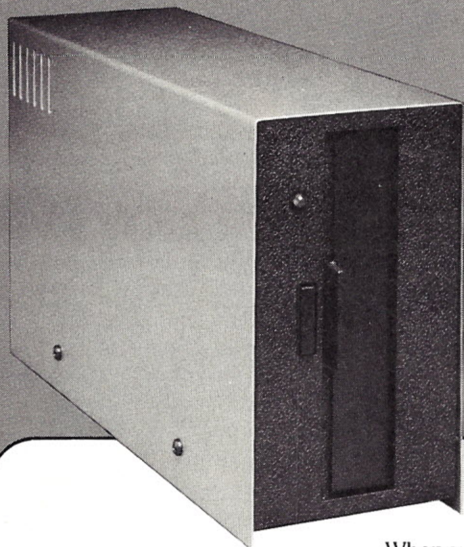


also:
Assorted Sorts
Business Forecasting Using Linear Regressions

ComputerCity

When the people behind the products count!

(Formerly the CPU Shop)



As the CPU Shop, we have been dedicated to meeting the needs of the microcomputer user. The success of the CPU Shop has led to ComputerCity—the merging of our manufacturing, wholesale and mail order divisions with our rapidly expanding retail outlets to provide the increased products and services the microcomputer consumers of today and tomorrow want—and need. We remain dedicated to providing the same service, technical assistance and fair pricing you've come to expect from the CPU Shop.

David C. Lourie, President

ComputerCity Sampler

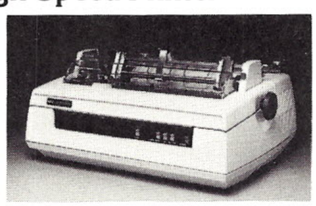
Disk Drives

When you're ready to add disk storage to your TRS-80*, we're here to help. Our CCI-100™ and -200™ drives offer more capacity than Radio Shack 35-Track (85K Bytes) drives. These drives are fully assembled, tested and ready to plug-in the moment you receive them. They can be intermixed with each other and Radio Shack drives on the same cable. 90 day warranty.

CCI-100™ 40 Track (102K Bytes) **$399.00** CCI-200™ 77 Track (197K Bytes) **$675.00**

Printers

Letter Quality High Speed Printer



NEC Spinwriter: Includes TRS-80* interface software, quick change print fonts, 55 CPS, bidirectional, high resolution plotting, graphing, proportional spacing and tractor feed assembly. 90 day warranty **$2979.00**

Also: Centronics, Paper Tiger, HI Plot Digital Plotter

16K Memory Up-grade Kits

Fast and ultrareliable	**$99.00**

DISK OPERATING SYSTEMS

NEWDOS by Apparat+	**$49.95**
NEWDOS "PLUS" by Apparat+	**$99.95**
DOS 3.0 by the original author of 2.1	**$49.95**

DISKETTE TRS-80*
BUSINESS SOFTWARE BY SBSG

Free enhancements and upgrades to registered owners for the cost of media and mailing. 30 day free telephone support. User reference on request.

Fully Interactive Accounting Package: General Ledger, Accounts Payable, Accounts Receivable and Payroll. Report generating.

Complete Package (requires 3 or 4 drives)	**$475.00**
Individual Modules (requires 2 or 3 drives)	**$125.00**
Inventory II: (requires 2 or 3 drives)	**$ 99.00**
Mailing List Name & Address II (requires 2 drives)	**$129.00**
Intelligent Terminal System ST-80 III:	**$150.00**
The Electric Pencil from Michael Shrayer	**$150.00**
File Management System:	**$ 49.00**
Budget Control Program II by CSA	**$ 49.95**
Cash Register System II by CSA	**$ 99.00**

ComputerCity

A division of CPU Industries, Inc.
175 Main Street, Dept. K-1 Charlestown, MA 02129

TO ORDER CALL TOLL FREE 1-800-343-6522
Massachusetts residents call 617/242-3350

Retail Store Locations:

175 Main Street, Charlestown, MA
K Mart Plaza, Manchester, NH
50 Worcester Road (Rt. 9), Framingham, MA
165 Angell Street, Providence, RI

Visa and Master Charge accepted

Franchise and dealer inquiries invited

Hours: 10AM - 6PM (EST) Monday - Saturday
For detailed information, call 617/242-3350
Massachusetts residents add 5% Sales Tax

™ CCI-100 and -200 are ComputerCity Inc. trademarks
* Tandy Corporation Trademark +Requires Radio Shack TRSDOS*

CIRCLE 3

JANUARY 1980 VOL. IV NO. 1

Personal Computing

For Your Home and Business

Page 55

Page 44

Page 64

Cover Design by Stephen C. Fischer

Cover Photograph by Alan Bergstein

Graphics on pages 55, 56, 59 & 60 courtesy British Tourist Authority, 680 Fifth Ave., New York, NY 10019. Copyright © 1980 British Tourist Authority.

LAUNCHING PAD

DIGGING IN

IN THE MONEY

DEPARTMENTS

Personal Computing

JANUARY 1980 VOL. IV, NO. 1

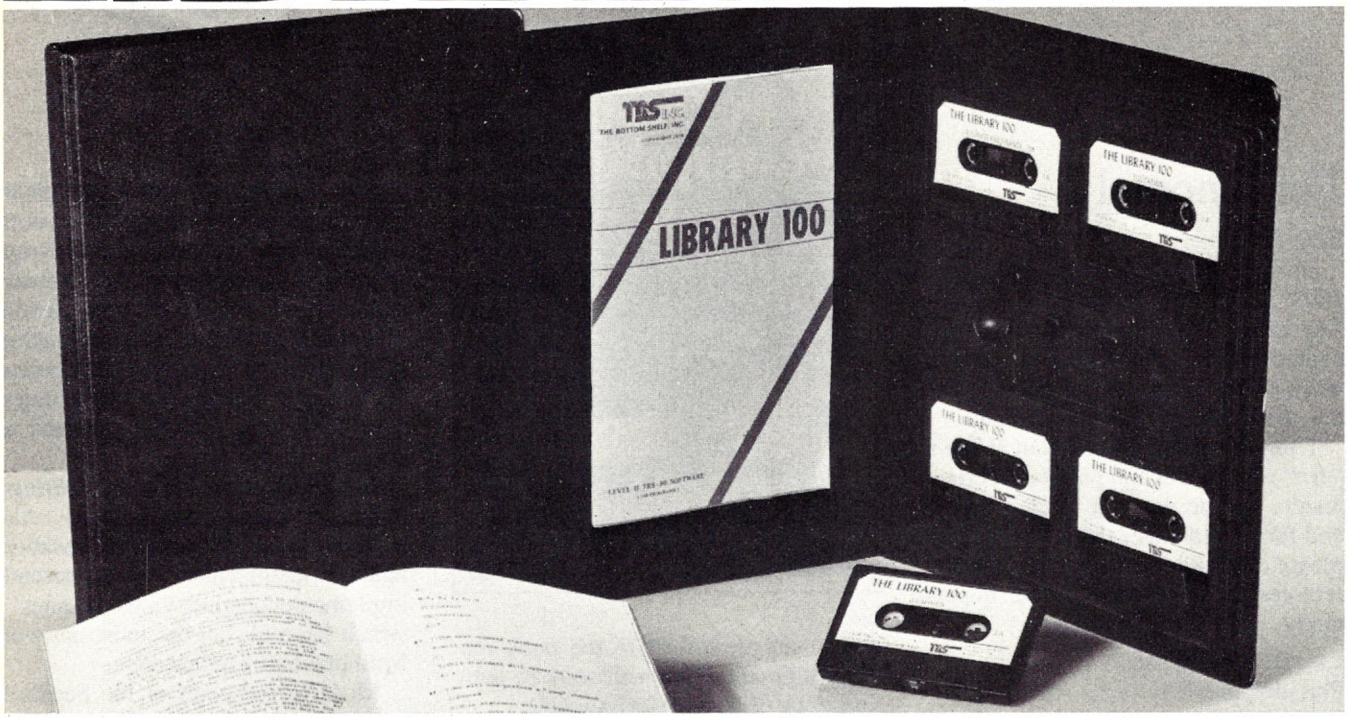
Publisher/Editor Harold G. Buchbinder
Managing Editor Don Wood
Associate Editor Harry Shershow
Assistant Editors Ken Mazur, Marjorie Morse
Art Staff David Bastille, Dolores Brandow, Stephen Fischer, Sue Genser, Diana Jenks, Josh Randall, Donni Richman, Susan Ross, Richard Sarno
Production Thomas Bronsky, Marlene Stibal
Advertising Production Jonna Yager
Composition Cheryl Dunphy, Julianna Crosier, Florence Daniels, Claudia Majetich, Maureen Riordan, Kathy Shank
Circulation Manager Dave Harriman
Circulation Pearl Ling, Regina Harrington
Bookstore Sales Martha Hawkins
General Adminstration Sarah Binder, Don Schaff, Sena Shaffer, Esther Shershow

Advertising Sales. Northeast: Arlene Joseph, Benwill Publishing Corp., 2 Park Ave., New York, NY 10016; (212) 340-9788 • **Midwest:** Pat Bosecke, P.O. Box 56089, Harwood Heights, IL 60656; (312) 631-7319 • **Northern California:** Ralph Petersen, 1380 Oak Creek Dr., #212, Palo Alto, CA 94034; (415) 328-4392 • **California:** Joanna Spiro, Mike Reynolds, 924 Westwood Blvd., Los Angeles, CA 90024; (213) 478-3017 • **Japan:** Hiro H. Irie, International Business Corp., 11-8, Narita-Higashi 1-chome, Suginami-ku, Tokyo 166; (03) 311-1746

Publisher. Published monthly by Benwill Publishing Corp.: Harold G. Buchbinder, Chief Executive Officer; George Palken, President; Domenic A. Mucchetti, Treas. Executive, Editorial and Subscription Offices: 1050 Commonwealth Ave., Boston, MA 02215. Controlled Circulation postage paid at Long Prairie MN. Membership in Audit Bureau of Circulation pending.

Subscription rates. U.S.: 1 year (12 issues) $14; 2 years (24 issues) $26; 3 years (36 issues) $38. Canada & Mexico: add $4/year for surface mail, $8/year for airmail. Other countries: add $8/year for suface mail, $36/year for airmail. Send subscription orders and address changes to: Circulation, Personal Computing, 1050 Commonwealth Ave., Boston, MA 02215. In Japan/Asia: Hiro H. Irie, International Business Corp., 11-8, Narita-Higashi 1-chome, Suginami-ku, Tokyo 166, Japan; Phone (03) 311-1746. In Europe: Personal Computing c/o LP Enterprises, 313 Kinston Road Ilford, Essex, Eng. IG1 1PJ tel: 01 553-1001. **Back Issues.** U.S.: $3. Canada & Mexico: $4. All other countries: $6.

LIBRARY 100



The **LIBRARY 100** from **TBS** is without doubt the greatest software bargain ever. Released in November 1978, it has sold thousands in 44 countries. Written for the TRS-80, **LIBRARY 100** contains 100 programs on five tapes. Most of the programs can be run on a 4K, Level II computer. Designed to be a basic computer library, it provides a series of programs over a broad range of topics. All programs but one are written in BASIC and can easily be modified to suit your own purposes.

"The program mix is eclectric, interesting, and curious . . . If I had a Level II TRS-80 and one or more grade-school children, or if I were a hardcore software collector, or if I had little software and wanted to get a lot of it with a minimum of bother, I'd buy the Library 100." *Stephen Gray, Creative Computing, April, 1979.*

". . . a basic computer library for the hobbyist, parent or businessman." *Kilobaud Microcomputing, December 1978.*

The programs are spread over five general categories; Finance, Education, Graphics, Home and Games. As an added bonus, the **LIBRARY 100** contains Tiny PILOT, a condensed version of the high level language primarily used in education. It is perfect for teachers, parents, students and sales trainees. Using only six commands, even a child could be programming in minutes. The other programs are as follows:

FINANCE: Present Value of Future Sum, Simple Interest for Days, Future Value of Present Sum, Amortization Schedule, Interest Rate-Compound Interest, Interest Rate-Installment Loan, Days Between Dates, Term of Installment Loan, Present Value of Series of Payments, Real Estate Investment Analysis, Nominal-Effective Interest, Internal Rate of Return, Future Value, of Regular Deposits, Regular Deposits for Future Value, Depreciation (Amount, Rate, Salvage Value, Schedule), Bond Present Value, Bond Yield to Maturity, Sale-Cost-Margin-Day of Week, Moving ad.

EDUCATION: Multiplication & Division, Addition, Subtraction, Fraction & Decimal, States & Capitals, States and Order of Entry, States and Date of Entry, States and Abbreviations, Inventors and Inventions, World Capitals & Countries, Urban Areas and Population, Authors & Books, Presidents and Order, States and Largest City, Base Numbers.

GRAPHICS: Front Cover, Wierd, Rat Race, Random Ad, Fireside, Left-Right Ad, Blocks, Herring, Launch, Blinker, Snoopy, Snow, Step Ad, Step Ad Two, Graphic Words, War Games.

HOME: Bartender, Nutrition, Conversion, Perpetual Calendar, Base Conversion, Calculator, Vacation Check-off List, Telecode, Message Board, Night Check-off List, Expense Account, Babysitter, Drunk-ometer, Remember, Christmas List, Mileage.

GAMES: Jumble, Search, Memory Quiz Letters, Sting Ray, Russian Roulette, Wheel of Fortune, Towers, Decision, Memory Quiz Numbers, Doomsday, Star Trek,™ Sketch, Flipper, Life, Fifteen, Speedy, Count, Road Race, Stars, Odd One, Spy Ship, Horse Race, Scissors, Craps, Star Blazer, Tiger Shark, Unjumble, Mind Reader, Roach Race, Jumble 2, Gypsy.

The price for the **LIBRARY 100** is only $49.50. That's less than $.50 per program. Join the thousands of users who are already enjoying this exceptional software package. Only from **TBS.** (We are currently working on a Library for the **APPLE**.)

TBS has other great software for your TRS-80. **CHECKBOOK II, INFO SYSTEM, & EXERCISER** are general applications. **BASIC TOOLKIT, SYSTEM DOCTOR & TERMINAL CONTROL** are systems utilities. **BUSINESS MAIL LIST, DATA MANAGER, CHECK REGISTER ACCOUNTING SYSTEM & ANALYSIS PAD** are strong applications for business. **TBS** also has **DISK HEAD CLEANERS** for **TRS-80** and **APPLE** and **GRAN MASTER DISKETTES**, the best on the market.

TBS is **YOUR COMPANY**, and to you we pledge to produce quality software at a price you can afford. The above products are available **NOW** at Computer Stores and Associate Radio Shack Stores nationwide or directly through us. For more information please contact us at the numbers below.

™ Paramount Pictures, Corporation.

TBS **THE BOTTOM SHELF, INC.**
(404)939-6031 • P.O. Box 49104-P • Atlanta, GA 30359

CIRCLE 7

FEEDBACK

Pet Anomaly

Dear Editor:

In a Feedback letter in the September 1979 issue, Lon Poole notes that his Day of the Week program (which appeared in the July 1979 issue) does not work correctly on the Commodore Pet due to an anomaly of Pet arithmetic.

I ran the program (without Poole's "fix") on my Pet and produced correct results for the dates in Poole's article and for several other dates. Would you please ask Poole to explain what incorrect results he got and, more importantly, what the anomaly is?

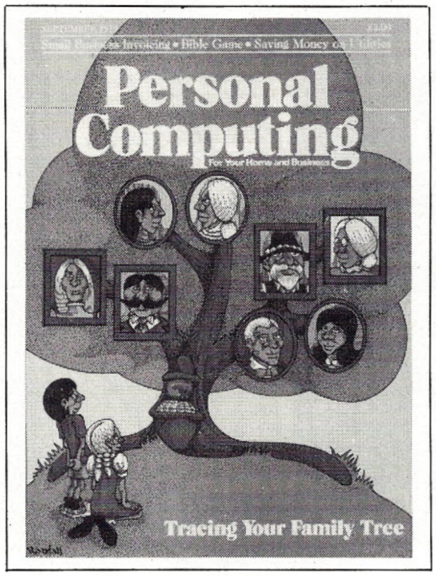
Charles I. Goldman
Philadelphia, PA

Author's note: The problem seems to be with the INT( ) function in conjunction with an apparent rounding error. For example, the Pet maintains that the expression INT((1E−20)−(1/.1)) does not equal the expression INT((1/.1)+(1E−20)). Furthermore, INT((1/.1)−(1E−20)) is greater than INT(1/.1). You can run the following program on a Pet to see this for yourself.

```
10   IF INT((1E-20)-
     (1/.1))<>INT((1/.1)+
     (1E-20))THEN 30
20   PRINT "EQUAL"
25   GOTO 40
30   PRINT "NOT EQUAL"
40   IF INT((1/.1)-(1E-20))
     >INT(1/.1) THEN 60
50   PRINT "LESS OR EQUAL"
55   GOTO 70
60   PRINT "GREATER"
70   STOP
```

I do not know why the Pet behaves this way, nor can I explain why adding the statement N=N on line 145 in "Day of the Week" fixes the error apparently caused by the use of the INT( ) function on lines 130 and 140. The error only manifests itself with a few dates. Unfortunately, I did not keep my notes on which dates don't work.

There are other Pet "features" which there is not room to describe here but



which are described in the book *Pet User's Guide* which will soon be published by Osborne/McGraw-Hill.

Lon W. Poole
Osborne/McGraw-Hill

POKEing Your PET

Editors:

One final comment on "Line Renumbering on the Pet" (March 1979): You can change the starting line number for the renumbered program by POKEing the desired value into memory locations 7259 and 7454. To change the interval between renumbered lines, POKE your choice into 7300 and 7484. All of these locations contained the decimal value 10 in the machine-language program as published.

Mark Zimmermann
Pasadena, CA

More on Copyrights

Dear Editor,

In response to the letter concerning software copyright infringement (Feedback, October 1979), I feel compelled to comment. I am not an attorney either, so these are only my opinions. However, I have had journalism classes, writing experience and photographic experience which has occasioned a familiarity with the copyright laws. I have followed this software copyright problem through several magazines and texts, up to and including, an editor's (not *Personal Computing*) request that readers should report their friends for infringement and receive a reward.

My feelings are that fear of infringement far exceeds the actuality. Also there are deterrents and compensations against infringement, most recently implemented against "music pirates", which keep all but the most unsavory from practicing in this area.

More to the point of the Feedback letter, how do we (the readers of *PC*) avoid unconscious infringement? Perhaps these questions will help others as they have helped me:

1. Is my program solving a similar problem using the equations, formula or statements as used in the original program?
2. Is the structure (arrays, data, jumps) of my program basically the same as the original?
3. Do my modifications have little or no effect on the flow, processes and/or results of the original program?

If the answer is "yes" to all or most of the above, then I have usually changed the application of the original program, not the program.

In closing, I want to thank *Personal Computing* for the fair and intelligent treatment of a very controversial subject. Your comparison of the copyright problems in prose was accurate and concise. Keep up the good work.

Howard Flatley
El Cajon, CA

Editor's note: If you'd like to express *your* opinion on the software copyright problem, write to Feedback, *Personal Computing,* 1050 Commonwealth Ave., Boston, MA 02215. We'll publish the best and most interesting letters. —*D.W.*

Computer-less Calculations

Dear Editors:

Bruce Barnett's article "How to Program a Complex Problem" (September 1979) provides a thoroughly understandable and enlightening example of the program development thought process. However, one doesn't need a fancy computer to figure probabilities for the game of Risk. Several months ago I calculated the relative advantages of various combinations of attacking and defending armies, all without the aid of electronic circuitry.

As the author stated, the probability that the attacker will lose 2 armies when rolling 3 die against the defender's 2 die is equal to the number of ways this can occur, divided by the total number of possible outcomes. Since a total of 5 die are rolled altogether, the denominator of this expression is simply 6^5, or 7776 possible outcomes. Finding the numerator is a bit more complex. Yet with only a quire of paper and several sharp pencils, I managed to derive a general formula for the number of different ways in which the attacker loses 2 armies. Without offering a mathematical proof (mainly because I have none), let me simply show you, in Figure 1, what the number of occurances happens to be.

Figure 1 $$\sum_{x=1}^{6} [(3x-2)(x-1)^2 + (6x-3) \sum_{y=x+1}^{6} ((x-1)^2 + 2(x-1)(y-x))]$$

Figure 2 $$\sum_{x=1}^{6} (-33x^3 + 262x^2 - 335x + 106)$$

Fortunately, $x-1$ can be factored out of the second summation, and after applying the formula for summation of arithmetic progressions and expanding the terms, the whole expression reduces to Figure 2.

Evaluating this expression for values of x from 1 to 6 produces the finite series 0 + 220 + 568 + 846 + 856 + 400, which equals 2890. Thus the probability of the attacker losing 2 armies is 2890/7776, or approximately 0.372, which agrees with the author's own finding.

Aren't you glad you do have a computer, Bruce? I sure wish I did!

David F. T. Kretzmann
Ft. Meade, MD

Aiding Visual Aids

Editor's Note: James W. Cerny, author of "Visual Aids for Business, Home and School" in the August 1979 *PC*, noticed a small inconsistency in the article's program. The message in the PRINT statement at line 1400 should read "changing lines 1120 and 1130 in the program" instead of "changing lines 170 and 180 in the program". Also, Mr. Cerny's affiliation at the University of New Hampshire has changed from Office of Academic Computing (line 1070 of the program) to the Computer Services department.

—*D.W.*

Hunt Sentenced

Editor's note: World Power Systems executive, Norman Henry Hunt Jr. is now serving a six year sentence for frauds committed by the bogus company, said J.W. Zemblidge, Postal Inspector in Tucson, AZ. Hunt entered guilty pleas to two counts of mail and wire fraud before U.S. District court in Tucson on September 17, 1979. His accomplice, Dinah Lee Hunt, pleaded guilty to a one count "information charging misprison of a felony" — a charge that she assisted in, but did not mastermind or pilot the fraud. Ms. Hunt received a three year suspended sentence and three years probation.

The Hunts, who worked as the husband and wife team of James and Lee Anderson, ran the bogus mail-order company out of Tucson. Federal officials called the scheme a "double bust out." World Power Systems sent out financial data to suppliers, ordered equipment on credit, but failed to pay the suppliers. The company also placed consumer ads, took cash orders from customers, then sent out only enough equipment to appear legitimate, according to the Pima County Attorney's Office in Arizona.

Hunt's swindle was first discovered through a WPS advertisement in which a picture of the company's I/O board showed no circuit etches. Hunt fled Arizona when he realized he had been discovered. Meanwhile, George Gerry Pollack, the man apearing in WPS ads, was arrested. Pollack has since been found an "investor's victim" and cleared of charges, said Zemblidge.

World Power Systems was not the first fraud created by Hunt, who perpetrated similar schemes in Nevada, Texas, California and Georgia.

—*M.J.M.*

New Pet City

To the Editor:

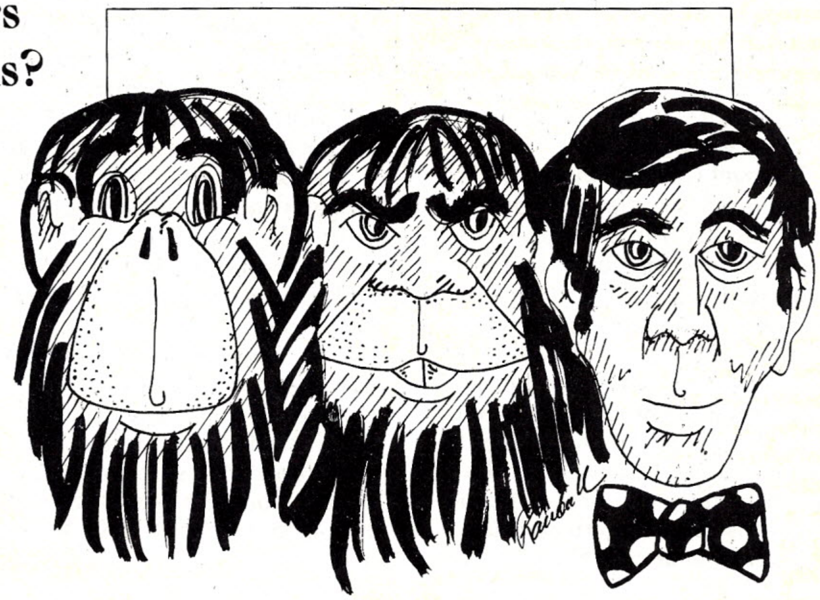
I'm not sending this short program into your magazine as a rebuttal to your published program called "City" (May 1979), but to broaden the horizons of all Pet programmers and to demonstrate the flexibility of the Pet computer.

Steve Kanzer
New Hyde Park, NY

```
10 DIM A$(7)
20 A$(1)="▚ ▀":A$(2)="%%%"
   :A$(3)="▚▚▚"
30 A$(4)="▒▒▒":A$(5)="@@@"
   :A$(6)="***"
40 PRINT "▐▌▌▌▌▌▌▌▌▌
   ▌▌▌▌▌▌▌▌▌▌▌▌▌▌▌▌▌▌▌▌▌▌▌▌▌"
50 FOR L=1 TO 4
60 FOR A=1 TO 8
70 B=INT(RND(1)*20)+1
80 O=INT(RND(1)*6)+1
90 FOR T=1 TO B
100 PRINT "▐███";A$(O);
110 NEXT T
120 FOR Y=1 TO B
130 PRINT "▌";
140 NEXT Y
150 K=INT(RND(1)*21)+1
160 FOR SS=1 TO K:
    PRINT "▌";:NEXT SS
170 NEXT A
180 PRINT "▐";
190 NEXT L
200 FOR T=1 TO 2500:NEXT T
210 GOTO 40
```

RANDOM ACCESS

Can Personal Computers Re-Create Missing Links?



Can personal computing systems trace evolutionary trends and recreate missing links in the earlier development of a species?

This startling idea grows out of work done by Professor D'Arcy Thompson over half a century ago. Although I have long-hoped to carry on Dr. Thompson's research, my own writing commitments have so far prevented this. If any reader would like to do some original, pioneering scientific research with his personal computer, here's an opportunity.

In his book *Growth and Form,* Thompson invented a novel application of curvilinear coordinate transformations. Thompson proved that a skull or body of one species drawn in a framework of curvilinear coordinates looks exactly like the skull or body of another species drawn in a Cartesian framework; and that the more closely-related the species (even if unrelated in appearance), the simpler would be the transformation. Thompson suggested that use of this method might provide a clue to the laws of growth in the evolution of species.

When Thompson drew the species of the genius *Diodon* — a fish rather typical in appearance (that is, looking like a fish should look) — on graph paper and then deformed the coordinates, something unusual occurred: the new fish (almost round in appearance) was now the sunfish *Orthagoriscus,* a fish known to marine biologists to be closely-allied to the former species. As Thompson explained: "I deformed its vertical coordinates into a system of concentric circles and its horizontal coordinates into a system of curves which resemble a system of hyperbolas. The old outline transformed in its entirety to the new network appears as a manifest representation of the closely-allied but very different-looking sunfish *Orthagoriscus.*"

With his curiosity piqued by this startling revelation, Thompson carried his research further. By mapping a human skull in a Cartesian framework, using it as a baseline and transforming it, he was able to recreate a chimpanzee's skull as a projection of the original rectangular system. Further transforming the coordinates, he produced skulls of other species. The more distantly removed was the species, the greater was the transformation needed.

Though he pursued the research even further, extending it to other animals, Thompson's research was to remain undeveloped — an avenue unexplored by modern researchers in archaeology, paleontology and the life sciences.

The significance of these interrelationships from a mathematical viewpoint now awaits further researching. Due to the extensive computational/graphical capabilities required, this research is well-suited to the personal computerist.

With the unfortunate situation of fossil remains identified for only portions of a species at certain fractional periods of its development, large gaps exist in our picture of past specifics. It's much like taking a snapshot of a certain species at intervals of time — sometimes millions of years apart — and then trying to piece its evolution together from this jigsaw puzzle. Furthermore, many fossil remains do not usually include the total skull or animal because portions are often missing. It is here that the interpolative power of the computer can be invaluable, since programs that can be generated will be able to interpolate between these known fossils, filling in the so-called missing links. It is also possible, with the proper research, that you may be among the first to accurately project evolutionary trends of species into the future.

This field could represent a totally new area of original research for the personal computer; and this is just one of many more applications that await those curious enough to pursue them.

— Paul Snigier

The author is editor of PC's *sister publication* Digital Design.

Cruise Line Enjoys Smooth Sailing

Champagne, gourmet meals, casinos, entertainment and sun make glorious and glamorous vacation cruises for travelers. But the logistics for planning cruises are not quite as glamorous for the weary cruise line.

Carnival Cruise Lines in Miami streamlined the planning process, saved several thousand dollars a year in operating costs, increased revenues by several hundred thousand dollars and improved office efficiency by installing a minicomputer system to book passengers, assign cabins, arrange travel to Miami, track payments, perform sales inquiries, handle special requests and record cancellations, said the cruise line.

A seven-year old, Miami-based firm with an annual passenger revenue of approximately $100 million, CCL books over 150,000 vacationers a year on seven-day cruises from Miami to San Juan, Puerto Rico, St. Thomas and St. Croix, Virgin Islands; Samana, Dominican Republic; and Nassau, the Bahamas. Individuals or groups may travel on the cruise and provide their own transportation to Miami, or book one of CCL's packaged trips: "Fly Aweigh" or "Fly Free", all inclusive air/sea programs which include airfare, ground transportation and the cruise.

Cruise line vessels include the Mardi Gras and Carnivale, both 1200-passenger ships, and the 1420-passenger Festivale. A fourth ship will be added to the line in December 1981 and possibly a fifth shortly thereafter, said CCL.

According to CCL's data processing manager, Jim Bussey, cabin assignments, payments and option expiration dates (the date cabin reservations are cancelled due to non-payment of balance) for each sailing were formerly recorded manually in berthing books. But because different departments are responsible for performing these tasks, the books were in constant demand. In addition, Bussey said, the information entered into the berthing books was hand-written, which added legibility problems.

Because the berthing book method was time-consuming and inefficient, CCL decided to find a better method of travel planning.

Galaxy World Tours, a Los Angeles tour operator working with CCL, told Bussey about their computerized travel system. Enthusiastic with their Data General Eclipse Data system, Galaxy invited CCL to see their computer in operation. Integral Business Computing designed and installed the system. "It wasn't a cruise system," Bussey noted, "but the basics were there, so we didn't have to look any further."

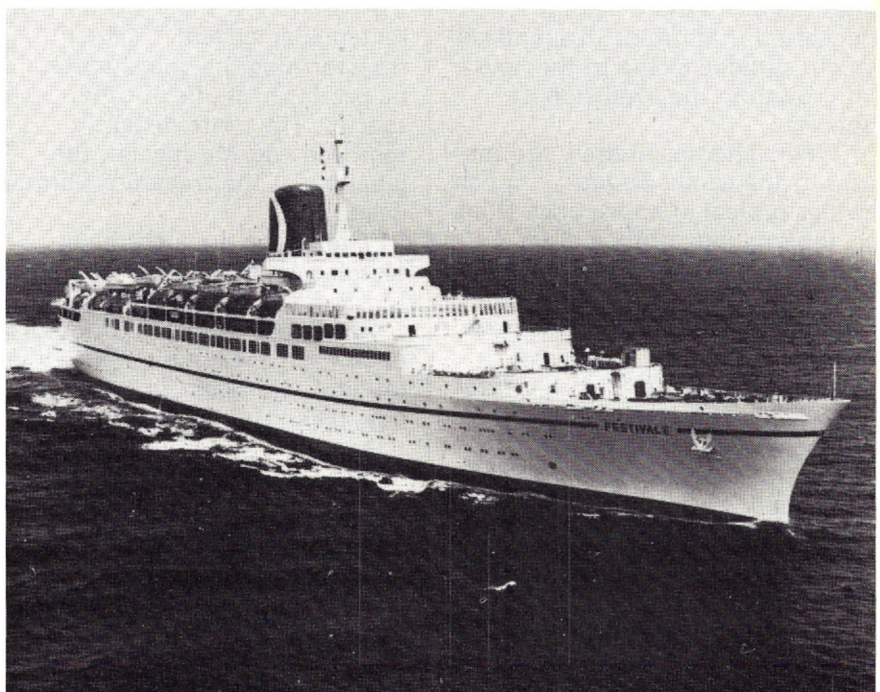
IBC modified their travel system to suit CCL's cruise application and installed an Eclipse C/330 with 512K bytes of memory, two 96-megabyte disks, one 10-megabyte disk, a 300-lpm printer, a 900-lpm printer and 35 Dasher display terminals at CCL's Miami headquarters. The system, operational since November 1978, runs on Data General's Real-time Disk Operating System. Programs are written in Fortran.

When an individual contacts a travel agent to book a cruise, that agent then calls CCL. Twenty reservationists use the Dasher screens to display "sailing availability" information, which lists the cabins available on the ship scheduled for the desired destination, along with cruise rate and cruise-plus-air rate. The travelers choose which cabin and which travel package they want.

A $100/person deposit reserves a cabin. Reservationists use the display terminals again to enter the traveler's names and assign a cabin number, and cashiers note the deposit payment. Line printers generate a reservation confirmation mailer and a deposit receipt which are forwarded to the travel agent. Groups of up to 100 people are handled in the same fashion.

Every detail pertaining to every passenger's trip is recorded in the system. CCL arranges air or ground transportation to Miami, limousine service to and from terminals, hotel accommodations, even car rentals, if requested. All of this information, stored in the Eclipse, can be referred to with a few keystrokes.

CCL also uses the computer system to inquire about sales.



Bookings are usually made six months to a year in advance of sailing. Formerly, it took a full week to count all these bookings one by one in the berthing books. Now it takes as little as two hours to determine how many passengers have been booked per ship.

In addition to confirmations, receipts and tickets, the computer printers generate a passenger manifest, which includes the names of all passengers booked, their cabin numbers, cruise ticket number and embarkation point. Previously, the berthing books were handed over to a service bureau four days before departure. The bureau would keypunch the passenger data listed in the books into a computer to prepare this report. But while the service bureau had the books for four days, CCL had to make do without them. A service report, also prepared by the computer, lists requests and special needs of passengers — a special diet, a bottle of wine, a wheel chair, or a cot.

Eye Banks Rely on Computers

The patient, a 41-year-old Atlanta accountant, had noticed the sight deteriorating in his right eye. He had the feeling he was continually looking through frosted glass.

An examination disclosed that the cornea — the transparent membrane forming part of the outer coating of the eye — was coned-shaped, a condition known as kerataconus, instead of round.

Fortunately, through the help of the Georgia Lions Eye Bank and Emory University's Computing Center, suitable donor tissue was quickly located in Houston and rushed by air to Atlanta. Timeliness in using new tissue is essential since the specimen can only be preserved for five days.

The corneal transplant operation performed in the Emory University Hospital was a complete success and the patient was able to read his wristwatch for the first time in months.

The Georgia Lions Eye Bank, operated in conjunction with the Emory University Department of Ophthalmology, is one of 18 eye banks throughout the United States now linked into a computer-communications network centered on the University's Sperry Univac 90/80 system.

The network started in January 1979 with 17 eye banks, according to Ron Wood, Coordinator of Information Systems and Programs at the Emory University Computer Center, who designed the program for the eye bank system. Since that time, additional eye banks have expressed interest in joining the network, Wood said.

Each eye bank has a terminal with dial-up capabilities and is linked to the computer system in the Computer Center over a 110-300 baud communications line.

Planning for the system started in the fall of 1978 when Wood was approached by the Georgia Lions Eye Bank to see if there was a method of locating cornea tissue in a faster and more economical way.

Janie Benson, Executive Director of the Georgia Lions Eye Bank, recalls the problems that led to the new computerized technique being introduced.

"Previously when we required tissue we usually had to make several long distance telephone calls to eye banks all over the United States before locating what was needed. It wasn't only time consuming but also expensive.

"Finally we felt that a computerized data base for a Tissue Inventory System (TIS) was the only answer. In response to our idea we received complete cooperation from Ron Wood and Dr. Buell Evans, Director or the Emory University Computer Center. A special program was prepared and the computer-communications network became a reality on January 8, 1979.

"For the most part the 18 eye banks in the system are operated by various Lions Club organizations in the different states in association with hospitals or medical schools in their areas," Ms. Benson noted.

In addition to Atlanta, the other eye banks are in Washington D.C.; New York City and Rochester, NY; Gainesville, FL; Iowa City, IA; Baltimore, MD; Columbus and Cleveland, OH; Dallas and Houston, TX; Portland, OR; Nashville, TN; Augusta, GA; Omaha, NE; Madison, WI; Richmond, VA; and Kansas City, MO.

When a cornea becomes available, key information is entered into the computer's data bank: name and address and 24-hour telephone number of the eye bank offering the tissue; the date and time the tissue was entered in the system; the type of tissue preservation; medical information about the donor — age, cause of death, data and time of death, date and time the tissue was removed, date and time of tissue preservation and the results of a slit lamp examination of the tissue. Each cornea tissue is assigned a number in the data base by the computer.

An Eye Bank entering information on donor cornea tissue is responsible for removing the entry from the system once it is placed. To keep the list current, tissue not removed from the listing by an Eye Bank is automatically removed by the computer after three days.

When located, the cornea tissue is packed in ice in a styrofoam container and shipped by air. Because of the demand and its short preservation life span, tissue never remains available very long in the data base.

About 350 cornea transplants are performed at Emory University Hospital and about 10,000 throughout the United States on an average each year. Since the computer network was organized, 75 transfers of tissue have been effected under the system.

Computers Provide Individualized Instruction

In an age of overcrowded classrooms, where the one-on-one student/teacher relationship has all but disappeared, over 6000 students at one community college have discovered that individualized instruction need not become a thing of the past.

With the help of 12 video display terminals housed in the school's library, Santa Ana Community College students in every department from computer science to physical education are reaping the benefits of personalized instruction.

"The terminals provide students with access to over 400 Computer Assisted Instruction programs in such diverse subject areas as psychology, biology, art, government and mathematics," explained Dean Strenger, Dean of Science and Technology. "These tutorial programs are designed to help students learn or review new material as well as overcome problem areas in certain subjects.

"If a student is having difficulty with a factoring concept in algebra, for example, he can call up a related program and the terminal will act as a tutor to the student, taking him through exercises which prompt skills and development in that particular area."

There is an entire menu of programs available to students ranging from a program which teaches how to play tennis to one which reviews art vocabulary.

"The programs serve as a review tool as well as an equalizer in classes where there is a broad level of ability," Strenger said. "In one business course, student grades improved 5 to 10 percent when a program summarizing class material was mandatory."

The terminals, which include nine Lear Siegler ADM-3 Dumb Terminal consoles and three ADM-1 terminals, were installed in the library about three years ago specifically for instructional purposes. They are being used on a Hewlett-Packard 2000 minicomputer. Previously, the college had been using teletypes, which, according to Strenger, "were expensive to maintain and very limited in scope for student needs."

In addition to the CAI programs, the Lear Siegler terminals are being used for other applications as well.

For example, computer science students in introductory computer classes are using the terminals to learn computer language and programming concepts. The terminals afford the students hands-on experience and provide a natural facilitation for what they are studying.

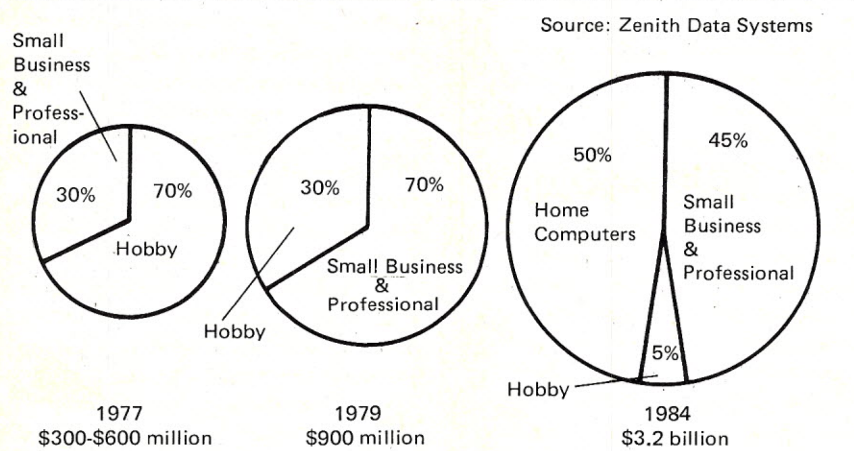
The department is also using the terminals for information retrieval purposes for students in computer assembly language classes.

Santa Ana Community College's system usage doesn't stop at academic applications. Students in a physical fitness class are also using the terminals to keep track of their weekly physical activities.

"In the beginning of the class, students are given a battery of tests to determine fitness," Strenger explained. "The tests measure flexibility, strength, lung functions, percentage of body fat and heart rate. A fitness program based on these test results is then worked out for each student.

"The student is responsible for earning a certain number of points each week by participating in various physical activities. Each type of activity, such as running, swimming, cycling and racquetball, is assigned a point value. Running an eight-minute mile, for example, would earn five points while playing an hour of racquetball would be worth 11 points.

"Each week the student enters his or her weekly activities into the terminal. The system responds by giving weekly and cumulative point and mileage



Reserach by Zenith Data Systems shows that the microcomputer market nearly doubled between 1977 and 1979. This growth is expected to continue, with the market increasing an estimated 350 percent by 1984. Charts above show the major segments of this market in the three years, 1977, '79 and '84, and the change that has taken, or will take, place there. Most significant, between 1977 and 1979, the small business and professional segment grew from 30 percent to 70 percent of the market. Sales of small computers for business and professional applications will continue to grow along with the market as a whole, but by 1984, home computers could emerge as the major market segment, with 50 percent of the business for microcomputers.

totals. This information is then passed on to the instructor for evaluation purposes. If there is a discrepancy between performance and recorded level of activity, the instructor can use the terminal to call up a detailed listing of an individual student's weekly records."

The college plans to develop and expand current applications as well as to link the terminals up with the mainframe computer in the computer center, said Strenger. This link-up would allow students to work in Fortran and Cobal in addition to BASIC.

Up, Up and Away: Launching the Space Shuttles

When the Space Shuttle program gets into full swing, up to 60 shuttle flights will be launched per year from the Kennedy Space Center at Cape Canaveral, FL, and the Vanderberg Air Force Base at Lampoc, CA. Although a long way short of the 5824 airline shuttles yearly between Boston and New York, the Space Shuttle is still a giant increase in space program launches. Fifty million dollars worth of Honeywell computers will handle those launches.

Up to three launches will be in preparation at each site at any one time. Two large computers, as well as an integrated network of minicomputers, displays, special hardware and software, work together to launch the shuttles. This combination, otherwise known as the Launch Processing System (LPS), is made up of the Central Data Subsystem (CDS) and the Checkout, Control and Monitor Subsystem (CCMS).

The Kennedy Space Center's CDS, consisting of two Honeywell dual-processor 66/80 systems, completed performance testing recently, and all support systems are go for the first shuttle launch scheduled for mid-1980. A similar Vanderberg installation has completed the first phase of a three-phase installation program enroute to its first launch in 1983.

Al Brobson from Honeywell explained that the CDS uses primary and secondary computers, each having its own operating system software but sharing main memory, peripherals and communications. Each performs different tasks, but should the primary system fail, a mandatory NASA requirement is that the secondary system flush its workload and take on the primary role within 10 seconds.

CDS configuration into two computer-support systems isolates direct shuttle-support tasks from non-direct tasks. The direct or primary tasks provide shuttle check-out engineers immediate, on-line access to information required to run tests and make flow-critical decisions. In other works, the primary CDS computer system provides dedicated on-line support to the CCMS's network of 60 minicomputers, displays and consoles.

The secondary CDS computer system provides support for those non-direct functions essential to monitoring the shuttle flow schedule. For example, simulation of ground-support equipment and shuttle systems for check-out; software development and training; on-line post-test data reduction and analysis; database management and information processing required by the time-critical flow schedule; console support in labs and engineering areas, and time-sharing support services for engineering.

The new distributed-processing approach to launching was developed to allow ground operations to be compatible with vehicle operations, projected launch rates and turnaround times, as well as meeting the economic objectives of the shuttle program. Some of the turnaround techniques used were borrowed from the airline industry. All LPS consoles are supported by a common large-scale processing capability represented by the Honeywell 66/80s. Each console computer supports a console group that commands and monitors certain functions; everything, for example, to do with propellants is controlled from one console group.

Computer Protects Art Museum

Rembrandt might have been confused by it all. Surely, El Greco would have been astonished. Undoubtedly, the great artisans of the past would have never imagined their works protected by a computer.

But, the Kimbell Art Museum in Texas does employ a small computer to control the museum's air environment, and a larger one for security in the 120,000-square-foot, modern complex.

Not only does the IBM Series/1 computer help preserve priceless art and artifacts by Rubens, Goya and others, it is saving energy and dollars, said a museum spokesperson.

The Kimbell building — a series of self-supporting cycloidal vaults of post-tensioned concrete — was designed by the late Louis I. Kahn, winner of the gold medal of the American Institute of Architects and the Royal Society of British Architects. In 1973 the museum won the highest award of the Illuminating Engineers Society, and the 1975 Honor Award of the American Institute of Architects, as well as top engineering and construction awards.

"Obviously, we're quite proud of the Kimbell — not only for what it houses, but of the house itself," said spokeswoman Shirley Spieckerman.

continued

The computer is used for power management/facility control, keeping inside humidity at a constant 50 percent, and the temperature at a steady 70 degrees. Precise control of temperature and humidity is critical for preservation of canvasses and the priceless oils painted on them.

Additionally, for security, a larger computer monitors by television public areas of the vast building and all doorways, said the building's supervisor.

Faster Service

You may get served faster than ever if the next cafe you dine at uses AM's Documentor, a new electronic management control system for restaurants.

The microprocessor-based system consists of a master keyboard unit, up to three remote dining room order entry terminals that let waiters electronically relay customer orders directly to food preparation areas, and tab and kitchen printers.

The system's enhanced 60K bytes of memory simultaneously handles order information from 40 waiters on 160 menu items and controls up to 200 open checks. Dining room personnel, using a special terminal key, can also call up information on 150 different wine selections without physically having to check wine cellar inventory, said the company.

Waiters start transactions by entering their employee and check identification numbers — along with the food order on a dining room terminal keyboard. As order items are keyed into the system, they are immediately distributed to printers placed in kitchen areas such as salad preparation, hot entrees or dessert.

Orders are printed out in the kitchen as received. Food items are then prepared and picked up by serving personnel. The Documentor master unit also computes prices, totals and prints out final checks when patrons have finished their meals.

The Documentor also gives management immediate quantity reports on up to 30 raw food items that a full service restaurant might carry in inventory. It can provide 35 other management reports — through software programs — including hourly sales, sales by food item, employee productivity and payroll time-keeping for100 employees.

☆☆☆Announcements☆☆☆

The Greater Baltimore Hamboree and Computerfest will be held Sunday, March 30, 1980, at the Maryland State Fairgrounds at Timonium. Personal and business computer exhibits along with dealers and hobbyists will fill the Exhibition Hall. Acres of space are available outside for tailgate sales and swaps. Many door prizes will be awarded throughout the day with hundreds of dollars in cash prizes given away at the grand drawing. Doors will open at 8:00 a.m. Admission is $3. For more information and reservations write to Joseph A. Lochte Jr., 2136 Pine Valley Dr., Timonium, MD 21093; (301) 426-8255.

International Society of Personal Computerists was organized to promote and advance personal computing on a worldwide basis. Tid-Bits, the Society's newsletter, is intended to be of broad general interest to users and hobbyists. The society also publishes several other more specialized newsletters. Other society services include free software, free consultation, custom programming, conversions from one BASIC to another and group discounts on software and hardware. Membership is $15. For more information contact International Society of Personal Computerists, 4554 Cristy Way, Castro Valley, CA 94546.

The Microcomputer Investor is a publication for investors using microcomputers. For an information packet, send $2 to: MCIA, 902 Anderson Dr., Fredericksburg, VA 22401. Back issues are now available.

The Medical Computer Journal, a publication of the Doctor's Club, contains information about the use of computers in the daily practice of the private physician. Each issue discusses one of the most common illnesses, a computer system, laboratory test interpretation and ideas for office improvement through the use of the computer. The major thrust of the publication is to bring the computer and the private physician together, and assist physicians in using computers to improve patient care.

Subscriptions are $15 per year. The subscriber will also receive the *Dr. Com Puter Report,* the *Medical Computer Journal's* newsletter. For information contact Dr. Aziz A Ghaussy, Editor, *Medical Computer Journal,* 42 East High Street, East Hampton, CT 06424; (203) 267-2934.

A Robotics Interest Group has been formed to spread knowledge among experimenters and in the robotics field. For more information send a stamped, self-addressed envelope to: G. Gregoire, 837 Bourbon Court, Mountain View, CA 94040.

The Marquette Computer Society of Marquette University offers membership to all students and faculty. For more information contact Marquette Computer Society, 610 North Seventeenth St., Suite 206, Milwaukee, WI 53233.

Build your own microcomputer as you learn computer technology at home.

New from NRI! The Most Complete and Up-to-date Home Study Course Ever Offered

As the microprocessor revolutionizes the computer world and microcomputers appear almost everywhere, NRI brings you a new, convenient, and effective way to keep up with this expanding technology. It's NRI's Computer Technology Course, created and designed exclusively for learning at home in your spare time.

Featuring NRI's Exclusive Dual Language Microcomputer

NRI goes beyond book learning to give you practical, "hands-on" experience in designing circuitry, interfacing components, programming, and troubleshooting. As you learn, you actually assemble NRI's designed-for-learning microcomputer, incorporating the latest advances in the state of the art. It looks and operates like the finest of its kind, actually does more than many commercial units. But NRI engineers have designed components and planned assembly so it demonstrates important principles, gives you working experience in detecting and correcting problems. And it's yours to keep, put to work in your own home or business.

You also build and keep your own test instruments, including a transistorized volt-ohm meter and CMOS digital frequency counter. And NRI's Discovery Lab® broadens your horizons with specialized experiments and theory demonstrations.

The Proven Way to Learn at Home

You don't have to worry with travel, classes, or time lost from work when you learn the NRI way. As they have for more than 60 years of teaching technical subjects, NRI brings the material to you. You study in your spare time, at your convenience, using "bite-size" lessons that program material into logical segments for easier assimilation. You perform experiments and build equipment using kits we supply. And your personal NRI instructor is always available for consultation should you have questions or problems. Over a million students have already shown the effectiveness of NRI training.

Choice of Courses

Several courses are available, depending upon your needs and background. NRI's Master Course in Computer Technology starts with the fundamentals, explores basic electronics and digital theory, the total computer world, and the microcomputer. The Advanced Course, for students already versed in electronics and general computers, concentrates on the microprocessor and microcomputer. In both courses, you build all instruments and your own computer.

Send for Free Catalog... No Salesman Will Call

Get the details on these exciting new courses in NRI's free, 100-page catalog. Shows all kits and equipment, lesson outlines, and full information, including facts on other electronics courses. Mail the coupon today and we'll rush your catalog. No salesman will ever call. Keep up with the latest technology as you learn on your own computer. If coupon has been removed, write to NRI Schools, Computer Department, 3939 Wisconsin Ave., Washington, D.C. 20016.

NRI Schools
McGraw-Hill Continuing Education Center
3939 Wisconsin Avenue
Washington, D.C. 20016

NO SALESMAN WILL CALL
Please check for one free catalog only.

All career courses approved under GI Bill.
☐ Check for details.

- ☐ Computer Electronics Including Microcomputers
- ☐ TV/Audio/Video Systems Servicing
- ☐ Complete Communications Electronics with CB • FCC Licenses • Aircraft, Mobile, Marine Electronics
- ☐ CB Specialists Course
- ☐ Amateur Radio • Basic and Advanced
- ☐ Digital Electronics • Electronic Technology • Basic Electronics
- ☐ Small Engine Repair
- ☐ Electrical Appliance Servicing
- ☐ Automotive Mechanics
- ☐ Auto Air Conditioning
- ☐ Air Conditioning, Refrigeration, & Heating Including Solar Technology

Name (Please Print) Age

Street

City/State/Zip

Accredited by the Accrediting Commission of the National Home Study Council

173-010

CIRCLE 9

FUTURE COMPUTING

Robots With Human Skills?

BY WILLIAM R. PARKS

The future development of computer technology hints at achieving artificial, human-like intelligence of unbelievable power. Perhaps some men and women living today will be around to experience a future shock when they are confronted by intelligent computer systems with artificial human voices and acute hearing mechanisms. These systems, they will discover, outperform their own human skills in areas of professional endeavors. In other words it's like the industrial revolution. Electromechanical devices outperformed feats of common human labor then, and caused whole categories of human occupational skills to disappear. Computers are destined to affect us in the same way.

With the coming "mind-in-the-machine" revolution I predict that certain white-collar and some professional-level jobs will disappear within one generation. Before discussing these jobs, consider, as an argument, an accepted test for artificial intelligence.

The test was conceived by A.M. Turing. According to Prof. Turing a computer is judged to be intelligent when you place a man at a terminal and ask him to communicate through the keyboard and CRT screen with whomever or whatever is at the other end of the line. The man at the terminal must decide whether there is a human at the other end of the communication link or whether he is communicating with a computer. If the man at the terminal can't make such a decision and if, in fact, he is communicating with a computer then that is artificial intelligence, said Dr. Turing. Common sense suggests that the Turing test is correct.

To repeat: when you can't be sure whether you are communicating with a human or with a computer — then you are in the realm of artificial intelligence. Has this experience ever happened? Yes! There are already documented cases of respected individuals who believe there is a group of people ("traffic controllers", if you will) who are intervening in interactive computing environments and are directing other people using the systems. There are even some cases of people who want to develop "private" interpersonal relationships with sophisticated "conversational" programs involving psychoanalysis. Imagine that!! Some people have been fooled into thinking that there is an emerging consciousness latent in some complicated or large scale computer systems.

Such a condition is not possible. However, we can see a natural development in artificial intelligence that will eventually produce highly sophisticated programs. These would be so complex and comprehesive that robot-like creatures might evolve that could act like humans, think like humans, and even show emotions like humans. But these artificial creatures will never be conscious of what they are doing. We, on the other hand, will be conscious of their actions and we might even attribute consciousness to them because of their great intelligent powers. However, this will be an obviously dangerous situation. It will be far healthier and safer to view artificially-intelligent machines as responding to programs input by humans. In other words, you will have to judge the experience with the same feeling you have when viewing the painting of a great master. Somehow you feel close to the painter — even if he had lived in a past era.

I like the analogy of cartoons. We attribute human-like traits to animals. This is a good mental exercise. Also, very entertaining. Never once do we seriously consider Bugs Bunny or Mickey Mouse to be living entities — unless we are very young children. In like manner, we can program computers to be highly personal machines that appear to be human and conscious. Another analogy is in the reading of a letter from a person we love or know very well. As we read the letter, we sometimes sense that the person is in our presence.

For these reasons, I think it is important to know the names of programmers. It is with *them* that we communicate when we engage in interactive computerized conversations. The most intricate and exciting letters one receives are really good programs written by human beings. It is *their* consciousness that we experience and not the computer's.

Needless to say, future evil forces could produce highly sophisticated computer systems that would use principles of artificial intelligence to control conscious human beings. The humans would be influenced by unconscious but highly intelligent robot-computers. In order to effectively defend against such evil powers we could develop equally sophisticated but intelligent *good* robot computers. Does this sound like science fiction? Good and bad artificial intelligences?

In summary, we can conclude that of all the areas of computer technology, artificial intelligence is the most powerful! It is bound to have the greatest impact on man. That leads us to define the areas in which artificial intelligence can imitate man's intelligence, perhaps even exceeed man's ability.

I think that the following occupations will be enhanced by artificial intelligence: programmer, teacher, medical doctor, clergyman, nurse, social worker, policeman, and any other similar occupations that require human interactions. These occupations are safe even in the coming world of robots. However, much of the labor itself done by these workers will be computerized! Occupations not directly related to socializing and conversations with people will be phased out. Robots can handle situations or activities that do not involve socializing such as: assembly-work in factories, repair of machines, certain categories of technicians, work in unhealthy environments. (Let's send robots into coal mines or in certain potentially dangerous chemical factories).

With artificial intelligence maturing as a well established field, perhaps, we might see unusual results. Most legal work, for example, will be done by computers and this will require the need for computer-oriented lawyers. This possibility could greatly reduce the need for training the many text-book lawyers that we do today. Other well-established professions could similarly be diminished in numbers. Teacher aides, bookkeepers, bank tellers, and others would be greatly reduced by computerization of their job functions.

Now You Don't Have to Wait for the Best

It's new. And it's a book that every microcomputerist should have. It's Volume I of **The Best of Personal Computing.**

Comprehensive check writing and accounting, stock market analysis, inventory control and vacation planning are only a few of the valuable computer programs contained within.

These programs, collected from two years of Personal Computing magazine, will run on most popular microcomputer systems.

Just using one computer program could easily be worth more than the $7.50 purchase price of the softcover 8-1/2″ × 11″ book of 196 pages.

Some of the other programs well received over the last two years and reprinted here include: Investment Analysis to help decide highest profit potential, Contract Labor Cost Accounting, Calculation of Seasonal Indices, Computerized Phone Directory, Use of Sensing and Control Devices in the Automated House, Personal Budget Control and Automatic Data and Appointment Reminder.

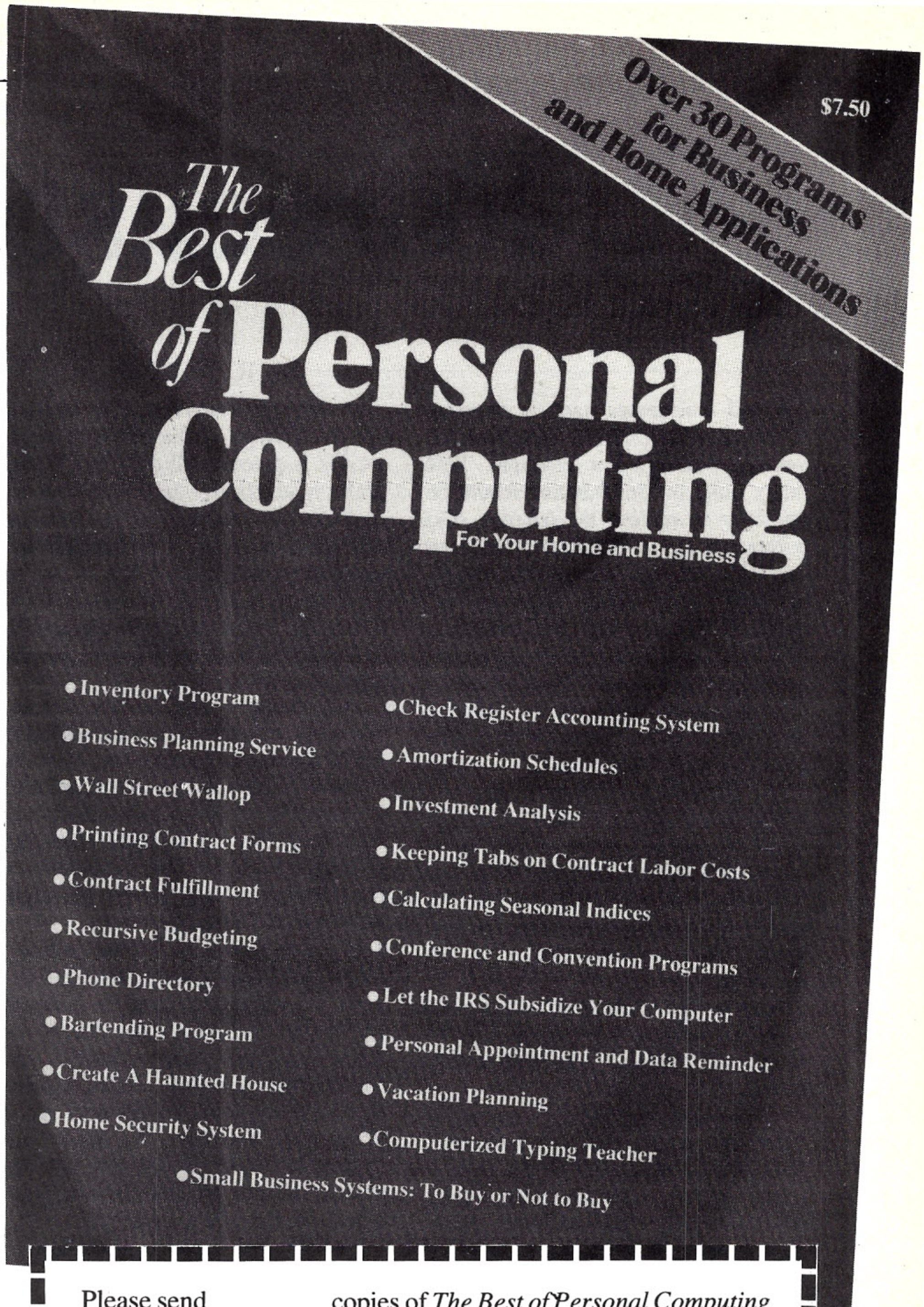
Programming students will find the program listings (mostly in popular BASIC programming language) an important aid to understanding and learning more about practical computer programming.

In total there are more than 20 applications with 30 program listings: 10 are applicable to both business and home use, 17 to business use, and 5 directly to home use.

One special feature of the book is the article "Small Business Systems: To Buy or Not to Buy", which takes the reader step by step through the questions he should ask himself, manufacturers and retailers before buying a computer system.

The Best of Personal Computing Volume I, is available for $7.50 (postage included) from **Personal Computing,** Box AD, 1050 Commonwealth Ave., Boston, MA 02215. Payment must accompany your order.

Treat yourself or a friend to a gift that will be appreciated for its usefulness — The Best of Personal Computing.



Please send ________ copies of *The Best of Personal Computing* at $7.50 each.

(Please print or type)

Name ______________________________

Address ______________________________

____

City ____________________ State ______ Zip ______

I have enclosed payment in the form of:

☐ Check ☐ Money order ☐ Master Charge ☐ Visa

Card Number

MC Only: Last 4 digits above name

Cardholder's Signature

Expiration Date

J1

Forecasting

BY R. TICKELL

Retail businesses face a classical forecasting problem involving inventory control. For example, how many units of merchandise should be on order to replace the quantity sold without being overstocked? Or, how can a retailer avoid the loss of business incurred when stock becomes exhausted? The problem can become a nightmare if the controlling influence behind volume of sales is not known or is improperly understood. Reliance must be placed on a forecasting system based on prior experience.

In forecasting, we take a series of numbers and guess what number will occur next in sequence. Later, we'll know what that next number really was and then guess the next one, using the experience gained in guessing the previous one and profiting by our mistakes.

Anyone can produce a forecast given sufficient information and the time to study it. But the task is simplified with the use of a computer.

The routine forecasting centers on "playing the averages" as best we can. Suppose, for example, we are given a series of five numbers: 180, 300, 360, 340, 220. What could we reasonably expect the next one to be? We might as well guess at the average value of the five numbers given — 280. Then, if the number really turns out to be 160, the question arises, "Could we expect to do any better on the next try?" We might, alternatively, use the average of the three most recent numbers (307). We could even adjust future forecasts on the basis of the error between the previous number forecast and the actual number reported.

The first two methods of forecasting are fairly conventional: the former is a simple average, the latter a moving average. The third method branches out in a different direction because it recognizes the possibility that a forecast depends both on the influence of an average taken from prior experience, plus an additional factor that is the amount by which the actual number differed from the forecast.

A serious disadvantage to using a simple average is the fact that equal weight is given to all data. We would prefer to emphasize the effect of recent data but not necessarily to the exclusion of prior experience. A well-established technique for achieving this means of forecasting is known as "smoothing", where short-term trends are recognized and weighed with the effect of longer-term experience.

But suppose that in the record-checking process we only had access to the most recent forecast; now, when the "actual" figure is announced and recorded we have only these two figures to work with. We probably don't feel very comfortable preparing a forecast for the next period, but here is what we might do to make an estimate.

It seems logical that, if the "actual" figure turned out to be higher than forecast, we should increase the estimate for the next period. Conversely, if demand is lower than estimated, we should lower the next estimate relative to the previous one. Furthermore, if the difference between "actual" and "forecast" is small, the estimate adjustment should be small; if larger, the adjustment should be sizeable.

This process leads to the following rule: To generate the new forecast, take the previous estimate and add to it a fraction of the amount by which the "actual" differed from the prior estimate.

Why not add all of the difference? This would be useful only if we know for certain that the trend is toward a continued rise (or a continued fall). Otherwise, the forecaster's prime concern should be to avoid large fluctuations or differences between "actuals" and "forecasts". This rule is expressed as:

(New Estimate) = (Actual − Old Estimate) × A + (Old Estimate)

(where A ranges between 0.0 and 1.0 at the discretion of the estimator).

Or, algebraically:

(New Estimate) = (Actual) × A + (Old Estimate) × (1 − A)

Using the larger values of 0.7 − 0.9 for A emphasizes the most recent experience while smaller values (0.1 − 0.3) diminish the effect of recent fluctuations.

Further consideration reveals that estimates computed by this "smoothing" method will lag behind even a steadily rising (or falling) trend. If a value can be assigned to the trend, then we can correct this tendency to lag behind what seems to be a steady, and therefore predictable, change.

An estimate of the trend is the difference between successive forecasts. Here, a weighting factor similar to the A factor can be used to correct the forecast derived from comparing the estimate and actual quantities. Call this Trend Factor "B", then use it to weight a trend correction as follows:

(New Trend) = (New Estimate − Old Estimate) × B + (Old Trend) × (1 − B)

A trend correction for the smoothed estimate derived above may be combined with it to produce a new forecast thus:

(New Forecast) = (New Estimate) + (New Trend) × (1 − B)/B

The computational simplicity of these terms is very attractive. The next set of figures are calculated using only estimate and trend from the previous period together with the "actual" reported — a useful simplification where record-keeping is concerned. For instance, consider the monthly inventory-management task of a warehouse carrying several thousand products. If each item has a history of monthly quantities-on-hand stretching back over a few years, the sheer volume of the record keeping task will represent a significant part of the business cost. The data in column 1 of Table 1 represents a series of monthly sales figures. Figures 1 and 2 display the limited effectiveness of simple averages when used to forecast sales figures over the same periods using data from previous periods. Figure 3 displays the results obtained with the "smoothing estimate" and from using optimized values of the weighting

factors A and B derived from a simple test program accompanying this article.

The accompanying BASIC was compiled and executed on a Sperry Univac 1108.

Here are a few questions you may ask before embarking on forecasting experiments.

- What are the optimum values for "A" and "B"? Some trial and error is necessary when you set up the system. But, as previously mentioned, the larger values of "A" lead to a more responsive or "nervous" system, as will be the case when smaller values of "B" are used. A simple set up of nested computations designed to test a large variety of combinations of "A" and "B" values can help determine a best fit before putting the forecasting procedure into service; this was done in the attached program.

- Should the values be fixed once and then maintained this way indefinitely? Not necessarily, but considering the foregoing reference to low values of "A" and the way they reflect long-term influences, it will take longer for a system altered from a higher to a lower value of "A" to adjust or settle down after a change than it will with a high value of "A".

- What values should be used for "old estimate" and " old trend" when setting up a new system? Use "actual" for the initial value of "old estimate". The forecast for the second period will be the "actual" from the first, thus:

$A \times (\text{actual}) + (1-A) \times (\text{actual}) = 1 \times (\text{actual}) = \text{new estimate}.$

This start-up problem will dissipate within a few periods. Trend should initially be set to zero and the system left to compute new values for itself.

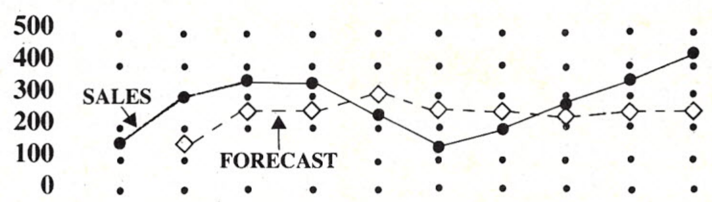
- Can the procedure be used for making forecasts for several periods? Not really. Consider what values should be employed to represent "actuals" for future periods. The best we could do is make them match the forecasts for the same periods. But doing this means that the principal input to the forecasting equation, ("A" times the difference between forecast and actual) would be zero. Then the Trend would become the dominant factor in the succeeding forecasts and, depending on whether "B" is greater or less then 0.5, would determine whether the system "runs away" or approaches a constant value; either way the forecasts are not of much use after two or three periods. Curve-fitting is a better prospect for long-term predictions, but that involves mathematical techniques beyond the scope of this article.

Table 1

Month	Actual Monthly Sales	Cumulative Average	Three Month Moving Avg.
1	180	—	—
2	300	180	—
3	360	240	—
4	340	280	280
5	220	295	333
6	160	280	307
7	200	260	240
8	280	251	193
9	360	255	213
10	420	267	280
11	—	282	353

Simulated sales data in column 1 used to derive estimates of future sales by simple cumulative average of prior sales (col 2) and by a three-month moving average (col 3); e.g. the quantity 280 in period 4 is derived from (180+300+360)/3.

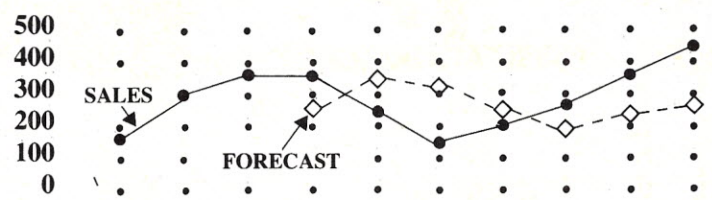
Figure 1



Graphic representation of data from columns 1 & 2 in Fig. 1. Prior estimate of the expected sales volume using a simple average of all prior months' sales.

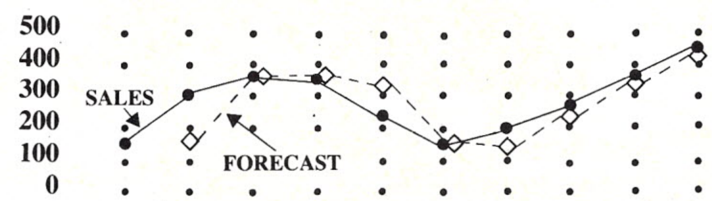
It can be seen that the average becomes increasingly insensitive to short-term changes due to the sheer weight of all prior data.

Figure 2



When the average value of the three most recent months' sales are used to estimate the expected sales for the following month it can be seen that to thus emphasize the most recent experience is merely to echo a three-month old sales history in the estimate of next month's sales.

Figure 3



Use of the smoothing estimate formula with A set to a value of 1.0 and a trend factor of 0.45 as computed and optimized by the accompanying BASIC program produced a notable improvement in the quality of forecast.

Program Notes

The accompanying program, written in BASIC on a Sperry Univac 1108, expedites the determination of optimum values for the weighting factors A and B in the smoothing formula for a given set of data. Once these values are established, the forecasting procedure becomes a simple three-step calculation (as in lines 320, 330, 340 of the program) for each new period using only the data available from the prior period.

Line 230 solicits up to 25 items of input data for testing for optimum values of A and B; more entries can be accomodated by adjustment of the DIM statement in line 210.

Lines 240 to 260 define initial values of "old estimate" and of "old trend" to allow a rational start to computation of the first forecast: S2 is a sum-of-the-least-squares quantity used to compare the quality of fit of various forecast curves tested during the optimizing exercise. The initial value set is suitably large to ensure that the first value computed is accepted for comparison with subsequent values when A and B are varied in the FOR loop (lines 270 to 460).

In line 300, the quantity NUM is a feature of Univac 1100 BASIC that automatically stores the number of data items entered in a MAT INPUT statement for the purpose of parameterizing subsequent operations involving the related MATrix. A simple counter can be devised to replace this capability if not otherwise provided.

The output PRINT statements are purposely written in an elementary form to avoid conflict with differing protocols under differing versions of BASIC. For example, the rounding algorithm INT(N*100+0.5)/100 looks cumbersome when compared with the various PRINT USING statements of most processors, since the latter usually provide built-in rounding. But, either way, the intent is to overcome the problem inherent in displaying decimal fractions of differing length that confuse the appearance of tables; for example, 0.4499998 as opposed to 0.45.

The MATrix function used in the program is a sophisticated operator. But the use of the function in this example is simply for storage of input data and derived lists of Trend and Forecast calculations. None of the elegant and powerful operations of matrix algebra are employed by the program.

Most BASIC processors support some form of storage wherein the elements may be referenced by an index mechanism such as a FOR loop or a counter. Such a collection of storage cells may be known as a matrix, an array or even as a vector, which is a single-dimensioned version of an array. Here we are, using vectors to store our lists of numbers: Vector D is the list of input data items that is processed to yield a list of related estimates, Vector C, with which the Trend calculations, Vector T, are combined to yield a reworked Vector C that is the list of Estimates we are seeking in the exercise. Lines 430, 440 in the program make exact copies of the Vectors C and T if the test conducted in line 390 is satisfied.

Line 210 may need modification to properly define the size of the five Vectors to suit the protocol of your particular BASIC. Line 230 might then be replaced with a simple input routine that assigns storage for up to 25 input data quantities and counts the actual number submitted (for later use as an index limit for operations carried out on the data). Lines 300, 360, 500 and 550 will also need to be modified, using the same index in place of the NUM quantity that is otherwise used to count the input items. Finally, lines 430 and 440 must be replaced with a small routine that loads a copy of Vector C into Vector Z and a copy of Vector T into Vector Y. □

Program Listing

```
00100 REM  FORECASTING PROGRAM THAT WILL TAKE
00110 REM  UP TO 25 INPUT DATA ITEMS AT  RUN
00120 REM  TIME AND GENERATE A FORECAST FOR THE
00130 REM  NEXT PERIOD. EXPONENTIAL SMOOTHING WITH
00140 REM  TREND COMPENSATION IS VARIED OVER THE
00150 REM  RANGE OF 0 - 1.0 IN STEPS OF 0.05 AND
00160 REM  THE BEST FIT DECIDED ON BASIS OF SUM-
00170 REM  OF THE SQUARES FIT
00180 REM
00190 REM  PROGRAMMER TICKELL JULY 25 1979
00200 REM
00210 DIM D(25),C(25),T(25),Z(25),Y(25)
00220 PRINT'INPUT HISTORIC DATA, EARLIEST
      FIRST (D1,D2,...DN)'
00230 MAT INPUT D
00240 C(1)=D(1)
00250 T(1)=0
00260 S2=10E10
00270 FOR B=0.05 TO 1 STEP .05
00280 FOR A=0.05 TO 1 STEP .05
00290 S=0
00300 FOR I=1 TO NUM
00310 N=I+1
00320 C(N)=D(I)*A+C(I)*(1-A)
00330 T(N)=(C(N)-C(I))*B+T(I)*(1-B)
00340 C(N)=C(N)+T(N)*(1-B)/B
00350 NEXT I
00360 FOR L=1 TO NUM
00370 S=S+(D(L)-C(L))**2
00380 NEXT L
00390 IF S2-S<0 THEN 450
00400 S2=S
00410 A1=A
00420 B1=B
00430 MAT Z=C
00440 MAT Y=T
00450 NEXT A
00460 NEXT B
00470 PRINT
00480 PRINT 'PERIOD','ACTUAL','FORECAST','TREND'
00490 PRINT
00500 FOR J=1 TO NUM
00510 Z(J)=INT((Z(J)*10+0.5)/10)
00520 Y(J)=INT((Y(J)*10+0.5)/10)
00530 PRINT J,D(J),Z(J),Y(J)
00540 NEXT J
00550 X=NUM+1
00560 PRINT X;TAB(30);
00570 PRINT INT(Z(X)*10+0.5)/10,
00580 PRINT INT(Y(X)*10+0.5)/10
00590 PRINT
00600 PRINT '  ALPHA=';INT(A1*100+0.5)/100;
00610 PRINT '  BETA=';INT(B1*100+0.5)/100;
00620 PRINT '  SUMSQ =';INT(S2)
00630 END
```

Sample Run

```
INPUT HISTORIC DATA, EARLIEST FIRST (D1,D2,...DN)
? 180,300,360,340,220,160,200,280,360,420

PERIOD        ACTUAL        FORECAST        TREND

 1             180            180             0
 2             300            180             0
 3             360            366            54
 4             340            393            27
 5             220            329            -9
 6             160            154           -54
 7             200            127           -27
 8             280            222            18
 9             360            324            36
10             420            404            36
11                            453            27

  ALPHA= 1    BETA= .45    SUMSQ = 39407
```

Linear Regressions for Small Businesses

BY MIKE DE SANTIS

Almost every facet of business, from predicting the Gross National Product to determining how much a theater should charge to show a movie, utilizes linear regression. Even though regressions are mathematically complicated, they're well worth the trouble to use — especially when your computer does the number-crunching.

Regressions mathematically mold data to form a model of a situation. A simple example would be the price history of a stock. By plugging data into the regression program you can determine what to expect the stock's price to be next year, thereby enabling you to decide whether to purchase the security. While regressions are usually more complicated than the example, often involving hundreds of variables and thousands of data points, let's look at the technique's use for small businessmen and students, whose needs are not so taxing.

The popularity of regressions goes hand-in-hand with the rise of computers. The complicated routines involved in performing regressions made hand calculations Herculean labors. Thus, only with the advent of computers did regressions become physically possible and economically feasible.

If you haven't already, look at the Program Listing. In the 107 lines of code, all the computer is doing is trying to fit a straight line like Y = AX + B through a set of data points (Y is the dependent variable, X the independent variable, A the estimator or slope, and B the constant term or intercept). The beauty of such a curve fitting procedure is that a good regression will give you a mathematical model on which to base business decisions.

Let's look at a concrete (but simple) example.

You serve as a part-time appraiser in your community. You've examined six lots of different frontages and values, and wish to make a model of this situation. The data is:

Value (Dependent Variable/Y)	**Frontage (Independent Variable/X)**
11,000 dollars	75.2 feet
8,800 "	58.0 "
12,200 "	81.0 "
11,100 "	69.0 "
10,500 "	72.0 "
13,000 "	85.0 "

Putting this data into the program will give:
Y = 357.58 + 146.42 X

You now have a mathematical model for the situation. If you wish to know the value of a lot with 110 feet of frontage, substitute 110 for X and find the answer ($16,463.89).

One of the nice things about regressions is flexibility. We could expand our model to include not only frontage but lot area (if the lots have different depths), distance from the center of town, tax rate and a host of other factors as well. Regressions such as these tend toward being an art form, allowing you the freedom to expand your model in any way you choose.

You may enter extra variables and data points to your heart's content, with one small catch: You must have one set of data points more than the number of variables for the program to work. That is, you can have 100 independent variables but you must have at least 101 sets of observations.

Of course, you can also run into trouble. By adding or deleting variables, you change the structure of the model; not only will the coefficients change but the accuracy of those estimators as well.(We'll see an example at the end of the article.)

To find the accuracy of the answers, statisticians use the "t-test."

The t-statistic is simply the coefficient (that is, the answer listed under the "Value" column) divided by the standard error of that answer. In general, a t value greater than two means the result is very good. If it is below two by a significant amount (say it is 1.7), you might want to discard that variable from your model. Simply re-run the regression without that variable.

Two more statistics are helpful in judging results: the F-statistic and the R-squared value.

The F-statistic indicates how good the model is as a whole. Thus, while the t-test will indicate how good an individual variable performs in the model, the F-test tells how good a relationship exists between your dependent variable (Y), and all the independent variables (the Xs). Here, the rule of thumb is 5. If the F value is above 5, you're OK. But if it is much below 5, you should scrap your model and try something else.

The R-squared value is related to the F-statistic but tells a

slightly different story; it indicates in percentage terms, how much of the variation among the data points the model explains. An R-squared of .92 means 92% of the variation is explained. Don't let a poor R^2 throw you off the track; different types of data yield different ranges of R-squared. A stock market model with a value of .60 is considered quite good because it is believed price fluctuations are due to psychological factors not readily quantifiable. A model of expected sales of an established product as a function of its price would probably require a R^2 considerably higher before it should be used.

I am only touching the surface here so experiment with the program. I'm sure you will find things that make your business life quite a bit easier.

Now, about the program itself.

I had difficulty devising this program because of the relative complexity of the problem. For those of you with some math, I'll describe the formulas used. If you don't understand, skip the material.

We have two matrices of data: Y, containing the dependent variables; and X (or in the program, the Ws), containing the independent variables. In matrix representation, the estimators of the regression coefficients (B) are determined by

$$B = (X'X)^{-1}X'Y$$

where primes indicate transposition and -1 means inverse. Thus, we have the problem of inverting a matrix. For this I took advantage of the relationship

$$X^{-1} = (1/DET(X)) * ADJ(X)$$

(DET = determinant, ADJ = adjoint).

This equation simplified matters because I already had a routine that took the determinant of a matrix. The routine transforms the matrix into an upper-diagonal matrix and the determinant is the product of the diagonal elements.

There is one command in the program that probably won't be available on your machine: NUM$ (line 6400). The command transforms a number into a character string. For systems without the same or a similar command, delete line 6400 and replace T$(I) in line 6500 by I. Just remember that your "1" will correspond to my "CONSTANT".

Line numbers 100 through 3000 set up the matrices for the regression proper, which is calculated between 3100 and 6000. The rest of the program contains both the subroutine that inverts the matrix and the commands to print the results. "CONSTANT" denotes that term in the model which I called "intercept" in my explanation. The other "values" are numbered in the same order as you instructed the program at line 2400. The Variance-Covariance matrix is necessary to find the t-statistics.

The program is slow but since most of you will use the routine on your own machine, CPU time does not cost you money. It should take between 5 and 15 seconds to perform a 2-variable regression on 20 data points. I do not know how much core space is needed. The program itself probably takes up on the order of 5K, with perhaps as much as 10K used by the many matrices. If you find it a tight squeeze, reduce the dimensions of the matrices.

When running regressions, put as much data as you can into the program. You could try to use three or four different dependent variables and one or two dozen independent variables allowing you to run several models at one sitting.

It would probably be useful if you saved your data on tape or disk. I didn't incorporate such a feature because I wanted to keep the program from getting too machine specialized.

Please, keep track of what the variables represent. No provision for names was made so you'll have to keep a separate record of what the variables mean.

Sample Run

Suppose you are in the business of manufacturing widgets. You have at your disposal the ACME Widget Making Machine. Like all machines, yours requires cash expenditure for maintenance and, like all machines, it will break down from time to time despite your best efforts. You would like to see just how the amount of money spent on the machine and the amount of time the machine runs each month affects the downtime of the unit.

In this case, downtime (in hours) becomes our dependent variable because it is the ultimate figure we want to find. Our independent variables are the maintenance costs and the number of man-hours the machine is used. We've gathered the following data for the last 11 months:

DT	HRS	MTN
59 hrs	976 man-hours	$1,000
72 "	603 " "	500
60 "	829 " "	750
71 "	838 " "	500
124 "	1,201 " "	450
95 "	1,149 " "	700
101 "	1,246 " "	650
75 "	1,054 " "	750
85 "	1,221 " "	800
103 "	1,182 " "	500
107 "	1,076 " "	450

Before we run the regression, common sense tells us to expect the sign of the coefficient for hours to be positive because the more the machine is used, the higher a downtime we should expect. The sign of maintenance is negative because the more money thrown into the unit, the less frequently it should break.

The first model I ran (See Sample Run) was:

$$DT = Constant + A_1 (HRS) + A_2(MTN)$$

From the printout, we see that the model is:

$$DT = 58.646 + .0757 (HRS) - .0786 (MTN)$$

Thus, our logic was right: the signs of the variables match what we thought they should be. The t-statistics, all far greater than 2 in absolute value, are significant, so all the coefficients should be kept in the model. The high F indicates the model as a whole is good; the R^2 of .885 means almost 89% of the variation in the data is explained by this model, which is quite good.

Suppose you know that in month twelve the machine will be operating 900 man-hours and you expect to pay $450 in preventive maintenance. Plugging into the model shows you should expect 91.41 hours of downtime and should make your plans accordingly. Similarly, if you desire no more than 80 hours of downtime in a 900 man-hour month, you should pay $595 in maintenance.

I said earlier that changing the model affects the accuracy of the estimators. In the next two regressions of the Sample Run, I show what happens when this is done. First, I ran DT = Constant + A_1 (HRS) with the result that the constant term is only 13.17. Notice that its t is .5 so we would throw it out of our model, leaving DT = .0709 (HRS). The hours t is now only 2.8, which is still significant but not by much. Although the F is in the safe zone, the low R-squared indicates you would not want to use this model to predict downtime. The last Sample Run using maintenance as the independent variable, has an even lower R^2. □

Sample Run

```
INPUT # OF OBSERVATIONS, # OF VARIABLES (N,K) ? 11,3

INPUT ROW #  1  ? 59,976,1000
INPUT ROW #  2  ? 72,603,500
INPUT ROW #  3  ? 60,8291312,750
INPUT ROW #  4  ? 71,838,500
INPUT ROW #  5  ? 124,1201,450
INPUT ROW #  6  ? 95,1149,700
INPUT ROW #  7  ? 101,1246,650
INPUT ROW #  8  ? 75,1054,750
INPUT ROW #  9  ? 85,1221,800
INPUT ROW #  10  ? 103,1182,500
INPUT ROW #  11  ? 107,1076,450

MENU

1=PRINT DATA MATRIX
2=CHANGE AN OBSERVATION
3=RUN REGRESSION
4=END
 ? 1

 59  976  1000
 72  603  500
 60  8.291312E+06  750
 71  838  500
 124  1201  450
 95  1149  700
 101  1246  650
 75  1054  750
 85  1221  800
 103  1182  500
 107  1076  450

MENU
```

```
 ? 2

ENTER I,J,X[I,J]  ? 3,2,829
```

```
MENU

1=PRINT DATA MATRIX
2=CHANGE AN OBSERVATION
3=RUN REGRESSION
4=END
 ? 3
```

```
WHICH IS THE DEPENDANT VARIABLE  ? 1
HOW MANY INDEPENDANT VARIABLES  ? 2
INPUT THE INDEPENDANT VARIABLES' COLUMN #'S
INDEPENDANT VAR. #  1  ? 2
INDEPENDANT VAR. #  2  ? 3

VARIABLE        VALUE           T
--------        -----           -
CONSTANT         58.64618        4.261356
 2               0.07570461      6.833902
 3              -0.07861646     -6.17648

R-SQUARED=  0.8855503
F-STATISTIC=  39.68731

MENU
```

```
 ? 3

WHICH IS THE DEPENDANT VARIABLE  ? 1
HOW MANY INDEPENDANT VARIABLES  ? 1

INPUT THE INDEPENDANT VARIABLES' COLUMN #'S
INDEPENDANT VAR. #  1  ? 2

VARIABLE        VALUE           T
--------        -----           -
CONSTANT         13.17355        0.5003132
 2               0.07095304      2.835253

R-SQUARED=  0.4130986
F-STATISTIC=  8.038637

MENU
```

```
 ? 3

WHICH IS THE DEPENDANT VARIABLE  ? 1
HOW MANY INDEPENDANT VARIABLES  ? 1

INPUT THE INDEPENDANT VARIABLES' COLUMN #'S
INDEPENDANT VAR. #  1  ? 3

VARIABLE        VALUE           T
--------        -----           -
CONSTANT         133.06          6.412688
 2              -0.07257599     -2.318329

R-SQUARED=  0.3043275
F-STATISTIC=  5.374579

MENU
```

```
 ? 4
```

Program Listing

```
00100 DIM Y(31,1),T(31,31),W(31,31),B(31,1),V(31,31),T1(31),T$(31)
00200 DIM W1(31,31),W2(31,31),W3(31,31),W4(31,31),Y1(1,31),C1(31),Z(31,31)
00300 PRINT \ INPUT "INPUT # OF OBSERVATIONS, # OF VARIABLES (N,K)"; N,K
00400 IF N<K-1 THEN PRINT "N MUST BE >=K-1!" \ GOTO 300
00500 PRINT
00600 FOR I=1 TO N
00700 PRINT "INPUT ROW # ";I;
00800 INPUT T(I,J); FOR J= 1 TO K
00900 NEXT I
01000 PRINT \ PRINT "MENU" \ PRINT
01100 PRINT "1=PRINT DATA MATRIX"
01200 PRINT "2=CHANGE AN OBSERVATION"
01300 PRINT "3=RUN REGRESSION"
01400 PRINT "4=END"
01500 INPUT Y2 \ ON Y2 GOSUB 1600,1900,2000,10700 \ GOTO 1000
01600 PRINT \ FOR I=1 TO N \ FOR J=1 TO K
01700 PRINT T(I,J);
01800 NEXT J \ PRINT \ NEXT I \ RETURN
01900 PRINT \ INPUT "ENTER I,J,X[I,J] ";I,J,Y2 \ T(I,J)=Y2 \ RETURN
02000 PRINT \ INPUT "WHICH IS THE DEPENDANT VARIABLE ";C
```

Continued

```
02100 INPUT "HOW MANY INDEPENDANT VARIABLES ";A
02200 PRINT \ PRINT "INPUT THE INDEPENDANT VARIABLES' COLUMN #'S"
02300 FOR I=1 TO A \ PRINT "INDEPENDANT VAR. # ";I; \ INPUT C1(I) \ NEXT I
02400 K1=A+1
02500 FOR I=1 TO 31 \ FOR J=1 TO 31
02600 W(I,J),V(I,J),W1(I,J),W2(I,J),W3(I,J),W4(I,J),Z(I,J)=0
02700 NEXT J
02800 Y(I,1),B(I,1),T1(I),Y1(1,I)=0 \ NEXT I
02900 FOR I=1 TO N \ W(I,1)=1 \ Y(I,1)=T(I,C) \ NEXT I  !Y=DEP VAR.
03000 FOR I=1 TO A \ FOR J=1 TO N \ W(J,I+1)=T(J,C1(I)) \ NEXT J \ NEXT I
03100   !READY TO RUN THE REGRESSION!!!
03200 FOR I=1 TO K1 \ FOR J=1 TO N \ W1(I,J)=W(J,I) \ NEXT J \ NEXT I
03300 FOR I=1 TO K1 \ FOR J=1 TO K1 \ FOR D=1 TO N
03400 W2(I,J)=W2(I,J)+W1(I,D)*W(D,J)
03500 NEXT D \ NEXT J \ NEXT I
03600 FOR I=1 TO K1 \ FOR D=1 TO N
03700 W3(I,1)=W3(I,1)+W1(I,D)*Y(D,1)
03800 NEXT D \ NEXT I
03900 GOSUB 6800
04000 FOR I=1 TO K1 \ FOR D=1 TO K1
04100 B(I,1)=B(I,1)+W4(I,D)*W3(D,1)
04200 NEXT D \ NEXT I
04300 FOR I=1 TO K1 \ W2(1,I)=B(I,1) \ NEXT I
04400 ! GET THE VARIANCE-COVARIANCE MATRIX
04500 FOR I=1 TO N \ Y1(1,I)=Y(I,1) \ NEXT I
04600 W8=0
04700 FOR I=1 TO N \ W8=W8+Y1(1,I)*Y(I,1) \ W3(1,I)=0 \ NEXT I
04800 FOR J=1 TO N \ FOR D=1 TO K1
04900 W3(1,J)=W3(1,J)+W2(1,D)*W1(D,J)
05000 NEXT D \ NEXT J \ W7=0
05100 FOR I=1 TO N \ W7=W7+W3(1,I)*Y(I,1) \ NEXT I
05200 S=(1/(N-K1))*(W8-W7)              !S=VARIANCE
05300 FOR I=1 TO K1 \ FOR J=1 TO K1 \ V(I,J)=S*W4(I,J) \ NEXT J \ NEXT I
05400 !FIND R-SQUARED,F-STAT...BUT FIRST GET THE MEAN OF Y
05500 Y2=0 \ FOR I=1 TO N \ Y2=Y2+Y(I,1) \ NEXT I \ Y2=Y2/N
05600 R1=(W7-(N*Y2^2))/(W8-(N*Y2^2))    !REGULAR R-SQUARED
05700 R2=1-(1-R1)*((N-1)/(N-K1))        !ADJUSTED R-SQUARED
05800 F=(R1/A)/((1-R1)/(N-K1))  !F-STATISTIC
05900 !GET THE T-STATISTICS...STAND. ERRORS ARE THE DIAGONAL ELEM. OF V
06000 FOR I=1 TO K1 \ T1(I)=B(I,1)/SQR(ABS(V(I,I))) \ NEXT I    !T-STATS
06100 PRINT     !START THE OUTPUT
06200 PRINT "VARIABLE","VALUE","T"
06300 PRINT "----------","----------","----------"
06400 FOR I=1 TO K1 \ T$(I)=NUM$(I) \ NEXT I \ T$(1)="CONSTANT"
06500 FOR I=1 TO K1 \ PRINT T$(I),B(I,1),T1(I) \ NEXT I \ PRINT
06600 PRINT "R-SQUARED= ";R2 \ PRINT "F-STATISTIC= ";F \ PRINT
06700 RETURN
06800 ! INVERT SUBROUTINE : INVERT W2
06900 ! FIRST FIND THE DETERMINANT OF W2
07000 N1=K1
07100 FOR I=1 TO K1 \ FOR J=1 TO K1 \ Z(I,J),W4(I,J)=W2(I,J) \ NEXT J \ NEXT I
07200 GOSUB 9000 \ A5=D0                !GOTO DET SUBROUTINE
07300 ! NOW MAKE THE COFACTOR MATRIX- FIND THE MINORS
07400 ! USE W2 FOR THE COFACTOR MATRIX
07500 N1=K1-1
07600 FOR I=1 TO K1 \ FOR J=1 TO K1
07700 I1,I2=1
07800 FOR R=1 TO K1 \ FOR S=1 TO K1
07900 IF R=I OR S=J THEN 8400           !CROSS OUT ITH ROW, JTH COLUMN
08000 Z(I1,I2)=W4(R,S)
08100 IF I2=N1 THEN 8300
08200 I2=I2+1 \ GOTO 8400
08300 I1=I1+1 \ I2=1
08400 NEXT S \ NEXT R
08500 GOSUB 9000 \ W2(I,J)=((-1)^(I+J))*D0      !COFACTOR MATRIX
08600 NEXT J \ NEXT I
08700 FOR I=1 TO K1 \ FOR J=1 TO K1 \ W4(I,J)=W2(J,I)/A5 \ NEXT J \ NEXT I
08800 ! W4 IS NOW INV(W2)
08900 RETURN
09000 ! DETERMINANT SUBROUTINE
09100 D0=1 \ FOR P=1 TO N1-1
09200 IF Z(P,P)<>0 THEN 9900
09300 FOR G=P+1 TO N1
09400 IF Z(G,P)<>0 THEN 9600
09500 NEXT G \ D0=0 \ GOTO 10600
09600 D0=-D0 \ FOR F1=1 TO N1
09700 D=Z(P,F1) \ Z(P,F1)=Z(G,F1) \ Z(G,F1)=D
09800 NEXT F1
09900 D0=D0*Z(P,P) \ FOR P1=P+1 TO N1
10000 IF Z(P1,P)=0 THEN 10500
10100 F2=Z(P1,P)/-Z(P,P)
10200 FOR F=P TO N1
10300 Z(P1,F)=Z(P1,F)+F2*Z(P,F) \ IF ABS(Z(P1,F))<=1E-4 THEN Z(P1,F)=0
10400 NEXT F
10500 NEXT P1 \ NEXT P \ D0=D0*Z(N1,N1)
10600 RETURN
10700 END
```

Assorted Sorts

BY DAVID GALEF

When you need an alphabetized list or sorted file of numbers, it isn't hard to construct a sorting program. But will the program be efficient and will it be right for the problem at hand?

Sorting has always been an extremely important computer function in business, where lists of names, files and code numbers play an increasingly greater part each year. Sorting became so important that it prompted research into a variety of techniques, each with differing strategies and varying efficiency. As a useful computing application, sorting originated with large computers for industry, rather than for the home. Nonetheless, sorting can be used for simple inventory, grading papers or building directories. No one has to dream up far-fetched home uses, since sorting fits thousands of practical applications already.

As an example of how sorting works, suppose you wanted to arrange a list of numbers with the smallest at the top of the list, followed by the second smallest, and so on. Quite a few sorting strategies will do the work for you, each with its own patterns and shifts. Unfortunately, the sorts easiest to comprehend and program are all too often those with the lowest degree of efficiency. Still, three of the most popular sorts for home use are simple and well worth examining.

You can use these sorting techniques for a variety of applications such as simple inventory, grading papers or building directories.

Possibly the easiest sort to implement is the effective, but somewhat inefficient, interchange sort. The strategy of the interchange sort appears in Diagram 1. Basically, it starts with the first two numbers on the list and if the bottom number is smaller than the top number, switches them. Then, the second and third entries are compared and switched if necessary; then the third and fourth, and so on. By the end of the list, the largest number has sunk to the bottom of the list, which has become partially sorted. The comparisons begin all over again from the top since each smaller number moves only one step toward the top in any given run. Sample interchange runs appear in Diagram 2.

Typically, the interchange sort may take up to n runs through a list of n items long (requiring $n-1$ comparisons), so that the efficiency is of the order $K \times n^2$, where K is about 1. Anyone familiar with the n^2 phenomenon knows how quickly the sorting workload can climb as n increases! A portion of the work can be saved by checking whether any interchanges have been made in each run through the list (the purpose of the flow chart's "out" flag). With no switches in the latest run, sorting can stop since all numbers are in order, although fewer than n runs were made.

A sort's order of efficiency refers to the worst case presented for that sort. In the interchange sort, the worst case occurs when the list is originally inverted with the largest number on top. For some sorts, the worst case is an already sorted list, which the sort proceeds to destroy before building it up again.

Despite its foibles, the interchange sort has the advantage of an internal or in-place sort. All sortings can be done by switching number locations without extra memory space.

A slight increase in efficiency, though still of n^2 order, can be achieved with another ordering technique called the bubble sort. The bubble sort, a modification of the interchange sort, gets its name, appropriately enough, from the strategy it uses. As shown in Diagram 3, bubble sort compares the first and second numbers just as the interchange sort does. With the bubble sort, however, the second and third numbers are compared and, if switched, then the first and new second number are compared. In this way, the third entry may become the first, finding its own level like a bubble rising through water. The fourth number is then "bubbled-up" — compared with the third entry; if switched, compared with the second, and so on. Diagram 4 shows a list being bubbled-sorted. By the time the last number bubbles up from the bottom, the list is completely sorted.

The most comparisons any entry undergoes before finding its own level is $n-1$. But any entry not switched after the first check remains where it is, never requiring further bubbling though it may be switched downward by another number moving up. Because some numbers may be bubbled all the way up from the bottom region, while others may require no bubbling, the average comparisons necessary along the one run of n numbers is about n/2, and often fewer. The order of efficiency is, of course, $K \times n^2$, since the worst case requires almost as much work as the interchanging sort's worst case. Still, the efficiency now resembles $n \times n/2$, so the constant K

is about 1/2. Bubble sort, an in-place process, is quite satisfactory for reasonably long lists.

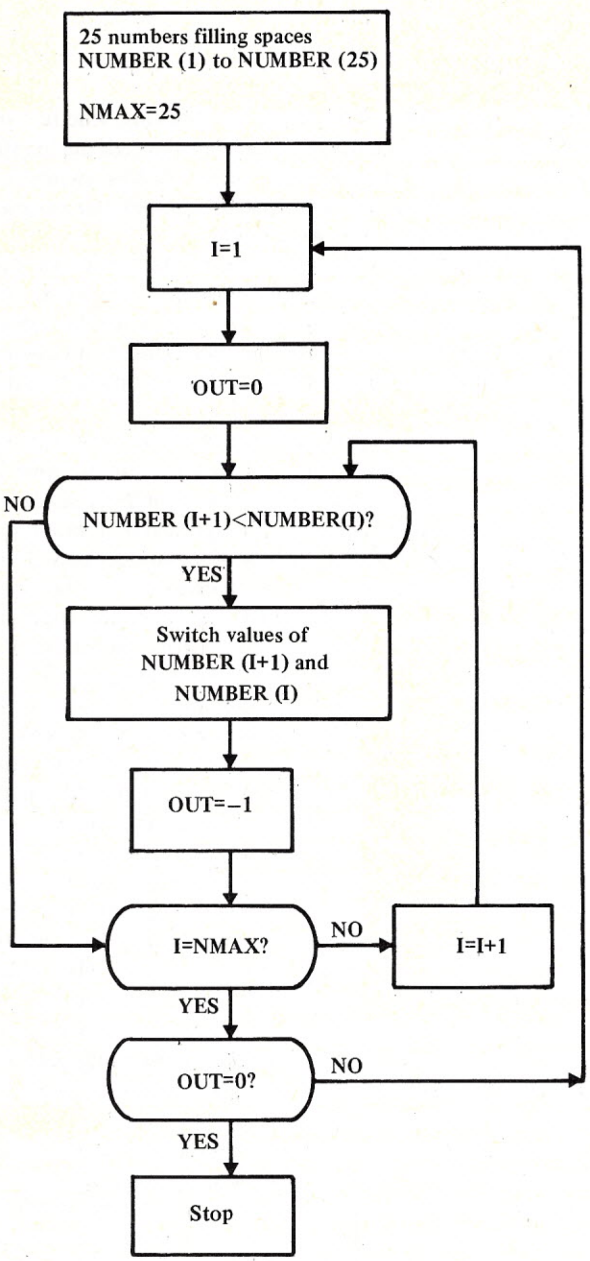
Another sorting procedure, the selecting sort, uses a comparison box to find the smallest number in the list on each run. As Diagram 5 shows, though the initial "smallest" number and its rank on the list are large dummies, the number and its rank will not stay in the SMALL comparison box for long. Every time a smaller number than the one in the box is encountered, the smaller number and its rank in the list replace the box's number and rank. At the end of a run through the list, the number in the box is taken off the list, or disqualified, with its rank on the list recorded. That number is then placed on a new, ordered list, which starts with the smallest number from the old list. The comparison box is filled once again with the dummy value — or the largest list number and its rank — to begin another search for the smallest number. To ensure that the same small numbers from previous runs are not chosen again, a flag will disqualify it from further consideration. In Diagram 6, the "out" flag for the particular number shifts from 0 to -1 when the number goes on the new list. The sort ends when the ordered list is complete. Because space for the new list must be set aside, the selecting sort is an external sort, requiring additional memory apart from the original list space. The order of efficiency is still $K \times n^2$, though decreasing comparisons for each successive run make it n runs of about n/2 comparisons: $K=1/2$.

Where the selecting sort differs from the previous two sorts lies in a modification that can be implemented to increase efficiency by decreasing the value of K. Merely break up the original, unsorted list into two columns of equal length, each with its own selecting strategy and comparison box. During each run, the smaller of the two numbers in the comparison boxes is put on the ordered list, and another run fills that box again. This strategy saves steps because each selecting sort is of order $(n/2)^2$, with n extra steps for the comparison between the two lists' comparison boxes. Thus, $n^2/4 + n^2/4 + n$ beats n^2 by a long shot, where n is large. Naturally, the process can be extended to breaking into three lists, each with n/3 items, for even greater efficiency: $n^2/9 + n^2/9 + n^2/9 + 2n$. As the number of small lists increases, however, the number of comparisons among various boxes becomes significant, and efficiency starts to drop. Then the programming can become quite messy.

There are, of course, other sorting procedures.

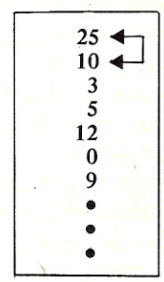
The digit sort works, not by ordering, but by sorting the numbers first according to their rightmost digits, then second-to-rightmost, and so forth. But rather than actually ordering each column of digits, wherein each column would

Diagram 1 – Interchange Sort

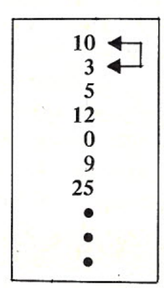


Initialization: NMAX is the number of entries; the comparisons start at the first entry, where I=1.
Interchanges are possible all through the run, until I=NMAX. Then, another run starts—unless OUT=0, indicating that no interchanges were made during the last run, and the list is sorted.

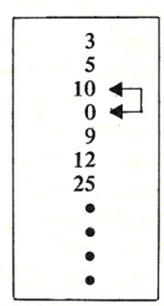
Diagram 2 – Interchange Sort



First interchange will be 10–25 switch, then 3–25. All interchanges in the run serve to bring 25 to the bottom.

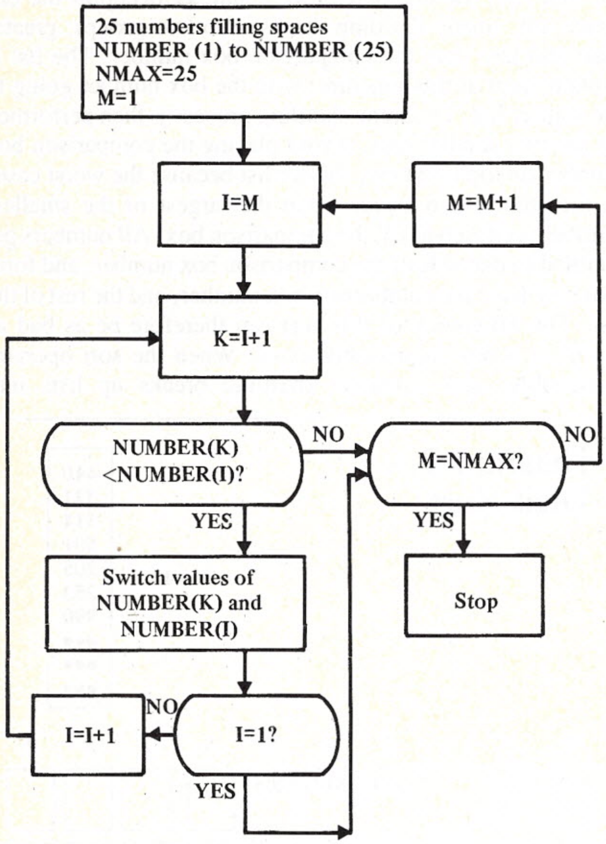


Next run: 10–3 switch, 10–5 switch, then 12–0 and 12–9 switch.



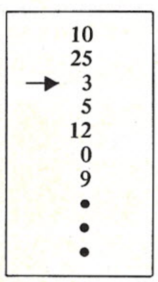
Third run: first switch is 10–0. Runs continue until sorting is over.

Diagram 3 – Bubble Sort

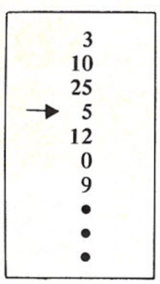


Initialization: NMAX is the number of entries; the comparisons start at the first (I=1) and second (K=I+1) entries. The marker M advances after each entry in the list is completely bubbled. A number is bubbled up until it reaches its own level, which may be at the top of the list where I=1. When all numbers have been bubbled, M=NMAX and the list is sorted.

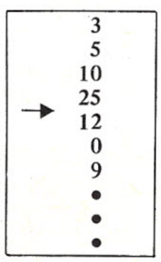
Diagram 4 – Bubble Sort



3 will bubble up past 25 and 10, stopping at the top.

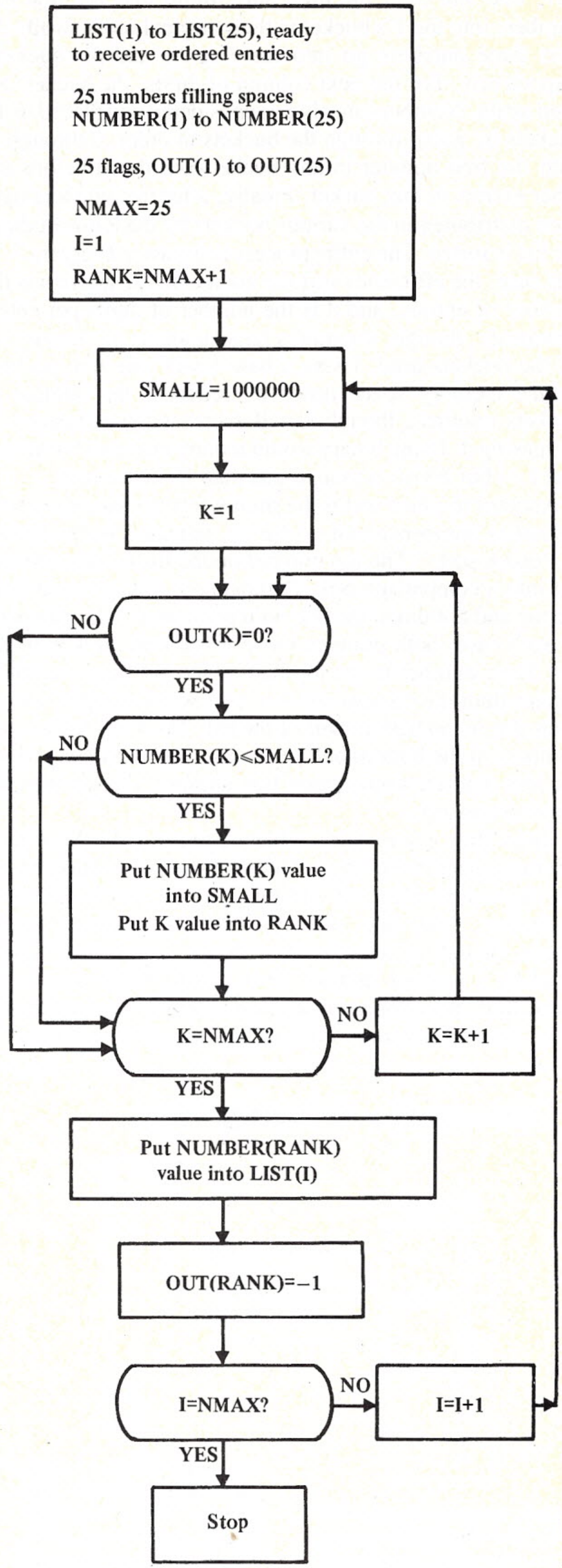


5 will bubble up to its own level, above 10 and 25, below 3.



Next number to be bubbled is 12. The sorting is over when each number has been bubbled.

Diagram 5 – Selecting Sort



Initialization: NMAX is the number of entries; each entry has its own OUT disqualification flag. Comparisons with the dummy SMALL value start at the first entry (K=1). I marks the progression of the ordered list. Each run, a new SMALL number is added to the ordered list; the sort is ended when the ordered list is full.

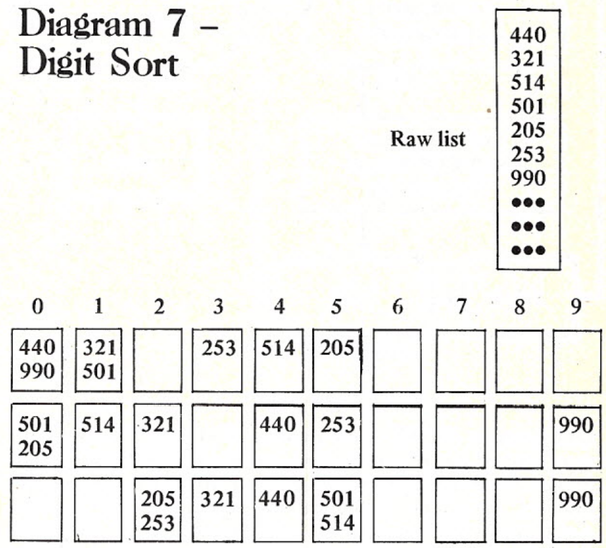
take many steps to sort, "buckets" representing each number from 0 to 9 are used. In the rightmost column for 391, 451, 567, 890 and 345, the digit sort would toss the first two numbers into the 1's bucket in their original order, while the next three numbers go into the 7's, 0's and 5's buckets, respectively. For the next column of digits, the buckets are used again. Numbers are always considered and placed in the buckets by going through the buckets in order, 0 through 9, with the first number in any bucket considered before any other entries in that bucket. Finally, after the numbers have been distributed in the various buckets by the leftmost digit, the list is sorted. The entire process is shown in Diagram 7. In this case, the efficiency is quite good at n×d, where n is the number of entries, and d is the number of digits per entry. Obviously, though, it requires large amounts of memory for all the buckets used. Using a base lower than ten can help reduce memory space needs by reducing the number of buckets required, though it will decrease the efficiency by increasing d. Using binary would require only two buckets, 0 and 1 — but 10010 is more digits than 18 in decimal.

Another in-place sort is the shuttle sort, so named because it shuttles numbers about on the list, though it is also known as "quick sort." The top number on the list is taken off and put into a comparison box. Comparisons then proceed from the top and bottom of the list, with pointers as place-keepers. In each case, both pointers move toward the middle of the list. Each number in the top half of the list (upper pointer) greater than the comparison box number, is slipped into an open slot in the bottom half of the list. The same follows for numbers in the bottom half of the list (lower pointer) which are less than the comparison box number, though they are shuttled upwards.

Open slots occur when other numbers move from their original places in the list, and the first slot opens when the top list number is taken off and put into the box at the sort's beginning. After that, there is always an open slot somewhere, though its location may change as one open slot is filled by a number shuttled up or down. When there is no immediate place for a particular number to be shuttled, the other pointer advances until a number in its half must be switched; then the numbers change places.

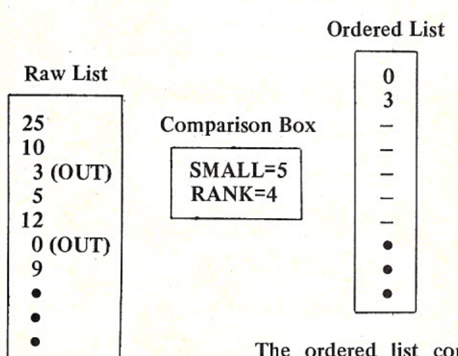
Diagram 8 shows a sample run. When sorting is finished for that run, the comparison box number occupies the last space left, neatly dividing the list into two groups: greater than and less than the comparison box number. The list is broken apart at this juncture, with the box number going to the shorter list. The same shuttling process is then performed on the two smaller lists. Avoid placing the comparison box number on top of a new, shorter list because the worst cases in the shuttle sort occur when the largest or the smallest number is chosen to fill the comparison box. All numbers get shuttled to one side of the comparison box number, and form two new lists: the comparison box number, and the rest of the list. The efficiency of this sort can therefore be as bad as $K \times n^2$ in a few improbable cases. When the sort operates reasonably, and continual shuttling breaks up lists into

Diagram 7 – Digit Sort



Buckets: in the first distribution, numbers are assigned buckets according to the 1's digit; in the next series, the 10's digit counts, and so on. In considering the numbers for buckets, the downward order from bucket 0 to 9 must be preserved.

Diagram 6 – Selecting Sort

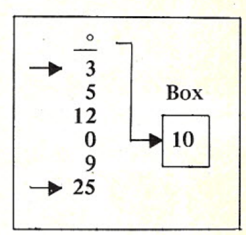


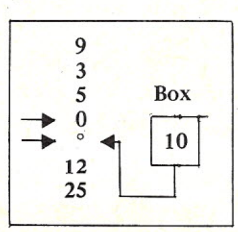
First number selected was 0 (then disqualified); next run selected 3 as the smallest remaining – also disqualified and put on ordered list. Next selection will be 5, of rank 4 (4th number), the number in the comparison box at the end of the run.

The ordered list contains the smallest number from the raw list on each run, stacked in order. A selecting sort with two lists and two comparison boxes puts the smaller of the two box numbers on the ordered list; the larger stays until it can be compared with a new number in the other box.

Diagram 8 – Shuttle Sort

Number on top of list, 10, is put in comparison box, leaving vacancy on top. Two pointers start at 3 and 25: no switches. At 5 and 9, 5 is less than 10 so it stays, but 9 on bottom is less than 10, so it is shuttled upward to original 10 vacancy. 12 on the top pointer is shuttled down to fill the vacancy; 0 goes to the 12 space. In each case, the bottom-pointer numbers must be less than 10 to be shuttled upwards; the opposite is true for top-pointer numbers.





End of a run: the numbers have been partially ordered through shuttling. The box number is put back into the open slot, a new box number is taken from the top of each new list (9, and a number other than 10 for the 2nd list). Notice that the two pointers meet at the end of each run.

Diagram 9 – List Ordered by Links

	Row	Number	Link
	1	5.4	7
	2	13.2	0
→	3	0.5	6
	4	7.6	10
	5	4.4	1
	6	0.9	8
	7	5.9	4
	8	3.2	5
	9	13.0	2
	10	7.6	9

Notice that the numbers are still in random order, but that the links establish an order. Starting at row 3, the next largest number is in row 6: 0.9. 0.9's link points to row 8, containing 3.2, and so on. The beginning of the link is marked by an arrow; the largest number has link 0 to indicate the end. Whatever the sorting used, the link structure involves no lengthy moving of data.

roughly half each time, an impressive order of efficiency results. Continual halving means binary efficiency, or $K \times n \times \log_2 n$.

Neither the digit nor the shuttle sort are all that easy to program. Furthermore, they share a serious limitation. Given an already sorted or nearly sorted list, both sorts will first aggravate the situation, then sort the disorder they have created. For example, a shuttle sort presented with a small-to-large ordered list would use the top number, the smallest, to order all the others.

Considerations such as the relative order of a list can therefore be quite important. Many people need sorting when their lists are only slightly out of order. For files with huge numbers of entries, other factors become important. The merging sort, for example, is used by many insurance companies not only because of its relative efficiency, but also because it can work by sorting parts of a file at a time — a necessary procedure with limited computer memory.

Some sorts, as they work, do not even switch around data. Though sometimes costly, this process establishes links from one datum to the next, making the order apparent — although the data are still stored randomly (see Diagram 9).

Perhaps the most efficient in-place sort is the heapsort, a technique employing an ordered binary tree with data on branches and a series of three-way comparisons. Because of the binary technique, efficiency is $K \times n \times \log_2 n$, even in the *worst* case. Heapsort, in all its phases, is on a rather sophisticated level in computer technology. When first developed, it was occasionally the subject of a computing science dissertation. Unfortunately, explaining the heapsort requires at least a good quarter hour and a blackboard.

Sorting thus becomes a sub-science in itself, which would, and has, filled a number of books. Certainly, earlier strategies, such as selecting and interchanging, are enough to start you learning. Possibly you can figure out a few modifications yourself. Think of the service you can provide. With a good sorting technique (or a few to meet different conditions) you can rent out your services to almost anyone. Merchants need mailing lists of customers, schools must have student classification and researchers want their statistics ordered. You probably have your own uses. The list is endless. All you have to do is sort it. □

Simple Software Sound Effects

BY J. ZDUNEK

After the first month or two of computer ownership, I decided to add extra hardware to my system, including a complex sound generator that could make dozens of realistic sounds. It received great reviews from those who heard it report explosions, screeching race cars and other selected noises.

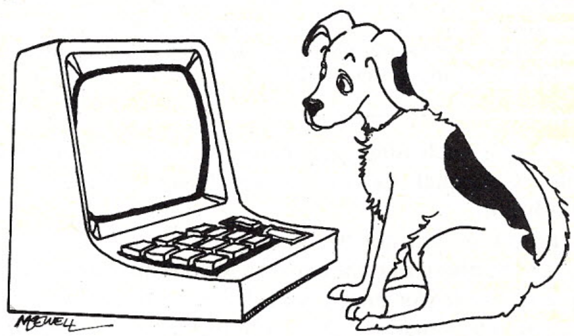
But, great as the sounds were, the disadvantages outweighed the seemingly unlimited uses of the sounds it would make. Being complex meant just that — it took days to get the right sound, and many times it refused to work properly in front of an audience. What I needed was a simpler way to get the job done — through software.

I looked back at some old computer music assembly programs I'd written for my 8080 microprocessor and rewrote them to fit my TRS-80. It worked off of an output latch on my home-built interface and produced two channels of square wave signals. Athough its quality did not compare with the complex generator, it was adequate for at least 70% of my needs and was worth keeping around.

One interesting thing about the operation of my new creation was that with a few minor changes in the program I was able to do away with the home-built interface entirely and use the cassette I/O circuitry contained in the TRS-80. Since the output circuitry could be borrowed from the cassette I/O, there would be no need to build any sort of address decoders or data buffers. Because of this, it would be easy to add sound effects programs to your TRS-80 by simply disconnecting the grey cassette AUX plug and plugging it into an amplifier-speaker system. Program 1 loads the proper machine codes that will activate the cassette I/O and produce a number of sounds and even music (of sorts).

You could use the program to produce a warble tone after every keyboard entry to indicate computer acceptance, or a low raspberry sound when data order is incorrect. Just about any game could be improved with sounds such as warp drive engines, phasors or pinball jackpot melodies. Special bell-type indicators could screech out and let you know when an error has occurred in the program. Over 64,000 different wave sizes could be produced without the extra hardware (except for the amplifier). The small amount of memory taken up by the software allows ample space for your main program.

The program requires a Level II TRS-80 with 4K memory and an external audio amplifier. The frequency generator, a short 45-byte machine language program, is easily loaded from a five-line BASIC program that pokes in the assembly listing. Once the routine is run, new programs can be loaded into the available memory space and edited to include the



audio output subroutine. The output is directed to the cassette tape output plug, which can be connected into an audio amplifier system to provide the necessary volume. Sounds simple? Try it and see.

Machine Set-Up Program

If you are converting old programs, I'd suggest saving this first program on the beginning of a new cassette and then transferring the edited programs on the balance of the tape. This procedure reduces future handling to one cassette. First, set the memory size to 20435 (see adjustments for 16K systems) and then run Program 1.

Program 1 – Machine Set-Up

```
10 DATA 0,0,245,197,0,0,0,14,1,0,0,62,1,211,255,
   6,1,16,254,0,0,0,62,0
20 DATA 211,255,6,1,16,254,0,0,0,13,32,231,0,0,0,
   193,241,201,0,0,0
30 FOR C = 20435 TO 20479 : READ E : POKE C,E
40 NEXT: FOR C = 20435 TO 20479 : D = PEEK(C) :
   PRINT D;
50 NEXT : PRINT "CHECK LISTING" : END
```

Be sure the data on lines 10 and 20 appear in the exact order as written, as they are the decimal codes for the machine language routine. When the program is run, the memory locations are rechecked and printed out for examination. If the data output is in the same order as the data input, type NEW and load any new or existing program. You can now add on subroutines that jump to the machine locations that make sound.

Try the following demonstration of a subroutine after you have loaded and run the machine set-up program. Connect

the grey output plug from the AUX input of the cassette recorder to an amplifier system (the home hi-fi does wonders), and run Program 2.

Program 2 – Demo.

```
10 INPUT A : GOSUB 3000
20 PRINT "ENTER" : GOTO 10
3000 ' WARBLE SUB (C) J. ZDUNEK 1979
3010 POKE 16526,211:POKE 16527,79:F=120:
     FOR R=1 TO 3 : FOR D =10 TO 15
3020 POKE 20443,D : POKE 20451,F : POKE 20462,F
3030 A = USR(0) : NEXT : NEXT : RETURN
```

With the amplifier properly connected, you should hear a warble sound each time you press enter. If the word ENTER appears each time, but no sound is heard, check the amplifier volume and connections. If ENTER does not appear, recheck the program listing — you may have entered the machine set-up program incorrectly and will have to reload the program from the start.

How It Works

Numerical time equivalents for the frequency "F" and the duration "D" are assigned in line 3010 of the demo program. In line 3020, the frequency and duration values are poked into the proper memory locations and then program control is transferred to the machine language routine through the USR(0) command in line 3030. The machine language routine is a simple triple loop that will activate digital highs and lows on the output latch in the TRS-80 cassette output circuit. These highs and lows are then amplified and converted into audible sound waves by the amplifier system. Program 3 is a listing of the machine code with Z-80 mnemonics. Note that the addresses and codes are in base 10.

Program 3 – Machine Code

Memory Location	Instruction byte	Mnemonic	Description
20435	0	NOP	(free
20436	0	NOP	space)
20437	245	PUSH AF	save flags and
20438	197	PUSH BC	A,B,&C registers
20439	0	NOP	
20440	0	NOP	(free space)
20441	0	NOP	
20442	14	LDCI, n	load DURATION BYTE
20443	1	(D)	into register C
20444	0	NOP	(free
20445	0	NOP	space)
20446	62	LDAI,n DURAT	load accumulator with
20447	1	(1)	bit #1 high and
20448	211	OUT, n	output to
20449	255	(255)	cassette.
20450	6	LDBI, n	load reg. B with f1
20451	1	(f1)	time byte
20452	16	DJRNZ,e HIGH	and hold output high
20453	254	(254)	for f1 duration.
20454	0	NOP	
20455	0	NOP	(free space)
20456	0	NOP	
20457	62	LDAI, n	load accumulator with
20458	0	(0)	bit #1 low and
20459	211	OUT, n	output to
20460	255	(255)	cassette.
20461	6	LDBI, n	load reg. B with f2
20462	1	(1)	time byte
20463	16	DJRNZ,e LOW	and hold output low
20464	254	(254)	for f2 duration.
20465	0	NOP	
20466	0	NOP	(free space)
20467	0	NOP	
20468	13	DEC C	decrement duration byte
20469	32	JRNZE	and jump back to DURAT
20470	231	(231)	if not equal to zero.
20471	0	NOP	
20472	0	NOP	(free space)
20473	0	NOP	
20474	193	POP AF	restore flags and
20475	241	POP BC	A,B,&C registers
20476	201	RET	return to BASIC program.
20477	0	NOP	
20478	0	NOP	(free space)
20479	0	NOP	

(Note that the free spaces can be used for making alterations)

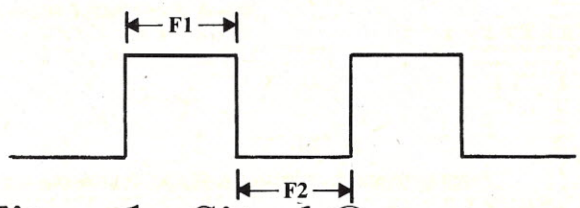


Figure 1 – Signal Output

Signal Output

The signal output in Figure 1 shows the square wave generated by the program. The length of the high signal is determined by the value of F1 (in memory location 20451); the length of the low signal is determined by the value of F2 (memory location 20462). The number of times one complete wave cycle is generated is determined by the duration "D" value of (memory location 20443). The actual duration or amount of time that a particular frequency is held constant must be corrected for time variance from F1 and F2 and would have to be calculated when specific musical notes are required. It is, however, not very critical for making ordinary sound effects "noise". See Program 4 for an example of how musical notes could be produced.

Restrictions and Alterations

Since the memory locations for F and D are limited to one byte each, their values can only be positive integers between 1 and 255. Add extra register loops in the machine listing if longer durations and lower frequencies are needed.

If you wish to load the machine listing on a 16K system, then the FORNEXT loops in lines 30 and 40 of the machine set-up program should read:

FOR C = 32723 TO 32767

Set the memory size at 32723; Poke the values for F1, F2 and D into locations 32739, 32750 and 32731 respectively.

Set the USR(0) function with POKE 16526,211 and POKE 16527,127 before using it in a subroutine.

With a little trial and error, you can produce a vast repertoire of auditory madness from your computer. I shall conclude with a familiar little number; try the routine in Program 4.

Program 4 – Tune Subroutine

```
4000 ' TUNE SUB, ARRANGED BY J. ZDUNEK
4010 DATA 130,130,115,100,130,100,115,180,130,130,
     115,100,131,136
4020 DATA 130,130,115,100,95,100,115,130,137,180,
     155,137,131,131
4030 POKE 16526,211 : POKE 16527,79
4040 FOR S = 1 TO 28 : READ F : IF F = 131 OR
     F = 136 THEN 4080
4050 D = INT((275-F)*.345)
4060 POKE 20443,D : POKE 20451,F : POKE 20462,F
4070 A = USR(0) : NEXT : RESTORE : RETURN
4080 D = INT((275-F)*.766) : GOTO 4060
```

When branched to the subroutine above, the program should play a little tune through the amplifier. I cannot guarantee a melody due to possible differences in your computer's clock frequency, but it's worth trying just for fun.

Students of music should be cautioned that the above program (which plays "Yankee Doodle") is a quick and dirty approach and should not be considered a serious piece of music. It does, however, have some use as an arcade game melody.

The two data lines contain the f values for each note to be played in the order as written. Note that the frequency of the note is used to determine the duration. For example, in line 4010, the first number is 130 and the second to the last is 131. Both f values are used for the same note although the frequency is slightly different. In line 4040, if the next f value read is 131, the note duration is calculated to be approximately twice as long by branching to line 4080. In other words, disregarding strict frequency relationships, note 130 is played as a "C" eighth note while note 131 is played as a "C" quarter note.

The duration calculations in line 4050 and line 4080 set the note as an eighth or a quarter and the value calculated for D is poked into the proper location. You can change the timing by adjusting the last term in the equation:

D = INT((275-F) * C)

where C is a number between 0.05 and 1.0 for any frequency f value from 20 to 255. (the f values 1 to 19 produce frequencies too high for my tastes.) You'll have to experiment to get additional notes required for your own use and add other branches to produce whole notes, half notes or whatever the particular melody you are making requires. □

f values: 180 155 137 130 115 100 95 180 131 136

Figure 2 shows the musical notation of the f values used in the TUNE SUB program. Note that 131 and 136 are quarter notes C & B, while values 130 and 137 are eighth notes C and B respectively.

NOW...
Exclusively Yours from ACS
TRS-80™-DOS 3.0
Disk Operating System
by the Original Author

☐ DOS 3.0
EFFECTIVE SEPTEMBER 1
ONLY $49.95

Check these DOS 3.0 features . . . that go far beyond 2.1

NO KEY BOUNCE • UPWARDS COMPATABLE FROM 2.1 • CHECK EOF
SEEK • RE-READ • RE-WRITE • LOC • VARABLE LENGTH RECORDS
SKIP • DISK LOGGING OF MESSAGES • BOOT • CHAIN • WRITE EOF
PURGE • SET • RESET • ROUTE • RUN & LOAD FOR 1 DRIVE SYS.
XFER • FORMAT W/O ERASE • DIR FROM BASIC • PATCH • LINK
USER DEFINED KEYS • KEY AUTO REPEAT • UPPER AND LOWER
CASE DRIVER • SHIFT LOCK • RS-232-C DRIVERS
• PAUSE • USER DEVICE CONTROL BLOCKS
• MULTI PROTOCOL COMMUNICATIONS

HARDWARE FOR TRS-80®

☐ **Pertec Disk Drives FD-200** .... $375.00 ea.
These are 40-track Drives that are completely compatible with the TRS-80™ and Radio Shack Drives. 3.0 DOS included. Will allow Turning Diskette over and Write on other side.

☐ **4-Drive Cable for Pertec Drives** $35.00

☐ **DECwriter III, 132 Character** $2500.00
110 to 9600 band EIA tractor feed keyboard printer. This is truly the nicest printer available. (30 day delivery)

☐ **701 Centronics TRACTOR FEED Bidirectional Printer** $1500.00
2½ times as fast as the Radio Shack 779 Printer, has full size 132 Char. Carriage Bell tone. Complete with Cable plug in and use. Shipped Freight COD.

☐ **200 ns 16K Dynamic Memory Clips for Keyboard or Expansion Interface, Lifetime Guarantee, complete** $110
Lifetime Guarantee. Complete with Instructions and Jumper Blocks.

☐ **10 Key Numerical Keypad Kit** $79.95

☐ **TRS-80™** Level II – 16k $750.00
☐ **Expansion Interface** $275.00
☐ **RS-232-C Interface** .... $ 89.00

ORDER NOW AND SAVE

Just list the items you want and mail this convenient coupon.

SOFTWARE BY ACS

☐ **Monitor No. 3** $29.00
Complete Machine Language Monitor for TRS-80™ features: Find, EDIT, Relocate, Symbolic Dump to Tape, etc.

☐ **Monitor No. 4** $49.00
All of the commands that reside in monitor No. 3, plus: RS-232 I/O, disk Program I/O, Symbolic Dump to Disk for Loading into Disk Editor/ASM., Track & Sec I/O for modification.

☐ **PCLEND** $15.95
Will Patch ASCII files of Basic Programs or text or DATA FILES so that they may be loaded into the Disk Version of the Electric Pencil for Editing purposes comes on Cassette that will automatically create a Disk file of PCLEND.

☐ **MAKE TAPE AND MAKE DISK for Cassette Dealers** $69.95
These are two programs that will allow you to take any type of Program from Disk and store it on tape for mailing purposes. When the µser receives the program in the mail on cassette, it is loaded into the computer which will automatically make a Disk file of the program.

☐ **CP/M & C BASIC for the TRS-80™**
CP/M Includes: MOVCPM, STAT, PIP, Dump, DDT, ASM (8080), ED, plus 6 user manuals.
CP/M $150.00
C Basic-2 Includes: XREF2, CBAS2, and manuals.
C BASIC 2 .... $99.95

☐ **G2 LEVEL III BASIC for TRS-80™** .... **Special $39.95**

☐ **TELCOM — Telecommunications for the TRS-80™** allows on TRS-80™ to communicate with another through the RS-232-C over the phone line.

™ TRS-80 is a registered trademark of Radio Shack, a division of Tandy Corp.

Orders received by 6:00 p.m. shipped next day on Master Charge, Visa, Certified Check or Money Order. Personal Checks require 14 days to clear. No C.O.D. Collect calls not accepted. All Hardware warranted for 90 days except Radio Shack equipment which is warranted through Radio Shack. Software guaranteed for replacement only. Prices subject to change without notice.

AUTOMATED COMPUTER SOFTWARE SERVICE
(615) 244-2798
Division of
Computer World INC.
625 Main Street • Nashville, TN 37206

Send Check or Money Order payable to —
SOFTWARE • P.O. Box 60097 • Nashville, TN 37206

Quan.	Description	Unit Price	Total

HANDLING CHARGE $1.50
TENN. RES. ADD 6% SALES TAX **TOTAL**

☐ Check
☐ Money Order
☐ MasterCharge
☐ Visa
Card No. ______
Exp. Date ______

Name ______
Address ______
City ______ State ______ Zip ______

Part 2

Calculator Accounting

BY MARLIN SNOW

Not only can the TI59 calculator do business accounting as discussed in Part I, but it can also be a time saver for tax accountants. Relevant data can be programmed and recorded on magnetic cards, eliminating the need to refer to tax tables. Taxable income is entered into Label A and the tax is displayed.

Before writing the program, construct a tax table like the one illustrated.

Tax Table

(1) Taxable Income	(2) Base Tax	(3) %	(4) Less
0 – 2,000	–	2	–
2,000 – 4,000	40	3	20
4,000 – 6,000	100	4	60
6,000 – 8,000	180	5	120
8,000 –10,000	280	6	200
10,000 –12,000	400	7	300
12,000 on	540	8	420

The first 3 columns represent a typical tax table. Column 4 is a correction factor to make programming easier. It reduces the number of steps required to program the table. Normally, the tax on 2,500 is calculated: 2,500 – 2,000 X .03+40 =. This method uses 18 key strokes. When the correction factor in column 4 is used, the key strokes are: 2,500 X .03 = –20 =. Key strokes are reduced to 14. Programming a small table does not require the use of column 4. However, if you have a long tax table with an extremely large number of tax brackets, the use of column 4 will enable you to get all of the program into the calculator. Column 4 is computed by multiplying the taxable amount by the %, then subtracting the tax as computed in the regular way. Using the example above, 2,500 X .03 = 75. The normal computation is 55. 75 less 55 = 20 — which is the correction factor for this bracket, (column 4).

The routine shown at right (with comments) is continued for each bracket until all of the tax table is programmed. The complete program is given in the Program Listing.

Program Location	Key No.	Key Symbol	Comments
000	76	LBL	Insert label A
001	11	A	
002	42	STO	Store taxable income
003	01	01	in memory #01
004	75	–	Less
005	02	2	2,000
006	00	0	
007	00	0	
008	00	0	
009	95	=	equals
010	77	GE	The amount in the register is compared to the amount in the T register (0). If the amount in the display is equal to or larger than 0, go to the next tax
011	00	00	bracket, memory location 021. If the
012	21	21	amount in the display is less than 0, compute the tax in this bracket.
013	43	RCL	Recall
014	01	01	memory 01
015	65	X	Times
016	93	.	.02
017	00	0	
018	02	2	
019	95	=	equals tax on amount entered in label A
020	91	R/S	Stop program. Tax is displayed.
021	43	RCL	Recall
022	01	01	memory 01
023	75	–	less
024	04	4	4,000
025	00	0	
026	00	0	
027	00	0	
028	95	=	equals. If the figure in the register is
029	77	GE	equal to or larger than 0, go to the next
030	00	00	bracket, location 043. If the amount in
031	43	43	the register is less than zero, compute the tax in this bracket
032	43	RCL	Recall
033	01	01	memory 01
034	65	X	Times
035	93	.	.03
036	00	0	
037	03	3	
038	75	–	less
039	02	2	20 (Refer to column 4 of tax table)
040	00	0	
041	95	=	equals tax.
042	91	R/S	Stop program. Tax is displayed.

Program Listing

```
000  76  LBL
001  11  A
002  42  STO
003  01  01
004  75  -
005  02  2
006  00  0
007  00  0
008  00  0
009  95  =
010  77  GE
011  00  00
012  21  21
013  43  RCL
014  01  01
015  65  ×
016  93  .
017  00  0
018  02  2
019  95  =
020  91  R/S
021  43  RCL
022  01  01
023  75  -
024  04  4
025  00  0
026  00  0
027  00  0
028  95  =
029  77  GE
030  00  00
031  43  43
032  43  RCL
033  01  01
034  65  ×
035  93  .
036  00  0
037  03  3
038  75  -
039  02  2
040  00  0
041  95  =
042  91  R/S
043  43  RCL
044  01  01
045  75  -
046  06  6
047  00  0
048  00  0
049  00  0
050  95  =
051  77  GE
052  00  00
053  65  65
054  43  RCL
055  01  01
056  65  ×
057  93  .
058  00  0
059  04  4
060  75  -
061  06  6
062  00  0
063  95  =
064  91  R/S
065  43  RCL
066  01  01
067  75  -
068  08  8
069  00  0
070  00  0
071  00  0
072  95  =
073  77  GE
074  00  00
075  88  88
076  43  RCL
077  01  01
078  65  ×
079  93  .
080  00  0
081  05  5
082  75  -
083  01  1
084  02  2
085  00  0
086  95  =
087  91  R/S
088  43  RCL
089  01  01
090  75  -
091  01  1
092  00  0
093  00  0
094  00  0
095  00  0
096  95  =
097  77  GE
098  01  01
099  12  12
100  43  RCL
101  01  01
102  65  ×
103  93  .
104  00  0
105  06  6
106  75  -
107  02  2
108  00  0
109  00  0
110  95  =
111  91  R/S
112  43  RCL
113  01  01
114  75  -
115  01  1
116  02  2
117  00  0
118  00  0
119  00  0
120  95  =
121  77  GE
122  01  01
123  36  36
124  43  RCL
125  01  01
126  65  ×
127  93  .
128  00  0
129  07  7
130  75  -
131  03  3
132  00  0
133  00  0
134  95  =
135  91  R/S
136  43  RCL
137  01  01
138  65  ×
139  93  .
140  00  0
141  08  8
142  75  -
143  04  4
144  02  2
145  00  0
146  95  =
147  91  R/S
148  00  0
149  00  0
150  00  0
```

As a rule of thumb, each tax bracket requires 30 program locations when the tax bracket is 1,000,000 and over. The federal estate tax table has 21 tax brackets from 0 to 5,000,000 and uses 553 program locations. The state death tax credit table has the same number of brackets and program steps, 554, for a total of 1,107. It is desirable to have both tables in the calculator because the federal tax is computed net of the state death tax. Since the maximum program steps that the TI59 can handle is 960, both tables obviously cannot be put into the calculator at the same time. However, this can be done by sending your programs to TI to be made into a custom library module.

Custom Libraries

If you have a TI58 or TI59 program especially applicable to situations where at least 500 potential users (sales forces, auditors, accountants, insurance agents, etc.,) share a common task, you may submit the program to TI to be tooled and produced. Your own library programs will then be put into custom modules similar to TI's standard library modules. (See "Random Access," Personal Computing, April, 1979)

Custom modules have the five following advantages:

1) A full 5,000 steps are available, as in standard libraries.

2) Common subroutines can be shared, making still more space available.

3) Solid-state programs cannot be accidently erased. And, they can be protected — preventing disclosure of proprietary information.

4) Several programs can be executed without having to handle magnetic cards.

5) Quick access to programs becomes available with the less expensive TI Programmable 58 rather than just the TI59.

TI58 and TI59 come equipped with a master library of solid state software programs contained in a small module inserted in the back of the calculator. This module contains 25 separate programs which can be called up by number, used individually, or incorporated into your own programs. In addition, you may purchase specialized modules such as statistics, real estate/investments, business decisions, etc.

TI offers the Professional Program Exchange (PPX-59) which makes software (developed by current TI59 owners) available to other users. By using PPX-59 as a vehicle to contribute and obtain programs, you will be able to increase the usefulness of your TI59 and broaden your professional base. □

PPX, an activity of Texas Instruments, was formed to provide useful services to its members. Dues are $18 a year-prepaid by MO or check. Members receive, as an introductory offer, three free programs chosen from PPX's catalog of 1700 different titles. Members also receive instructions on how to write and submit programs for the PPX. Acceptable programs are purchased by PPX, then listed in the catalog. In addition, members receive an informative newsletter 6 times a year and the exchange serves as a source through which accessories may be ordered. The address is TI; PPX Dept.; PO Box 53; Lubbock, TX 79408. Tel. no.: 806-741-3277.

Merging on the Pet

BY DAVID MULDER

Did you ever write two good programs and realize they would work great together as one program, or write a short program that would make a good subroutine in a larger program, but have no easy way to merge them? Or even more aggravating, have you ever wanted to use half of one program and half of another to produce a new program? To merge the two, you have to load one program, destroy all the unwanted lines in that program, and then type in all the wanted lines from the second program. And with the Pet keyboard, the more typing you can avoid, the better.

All these problems prompted me to sit down and figure out some way to merge all or part of two or more programs into one.

Right away, I decided that it was unlikely I would be able to disassemble BASIC and find the secret to the mystery. So I put together some of the things I knew that I thought might be useful. First, I knew that memory location 525 contains the number of characters in the keyboard buffer. Also, locations 527 to 536 contain the actual characters in the buffer. By changing the contents of these addresses, you can fool the computer into thinking you've pressed keys when you really haven't.

I decided I could use the keyboard buffer along with a tape file to print lines of programming on the screen and then set the computer to do carriage returns on these lines, entering the lines into the new program.

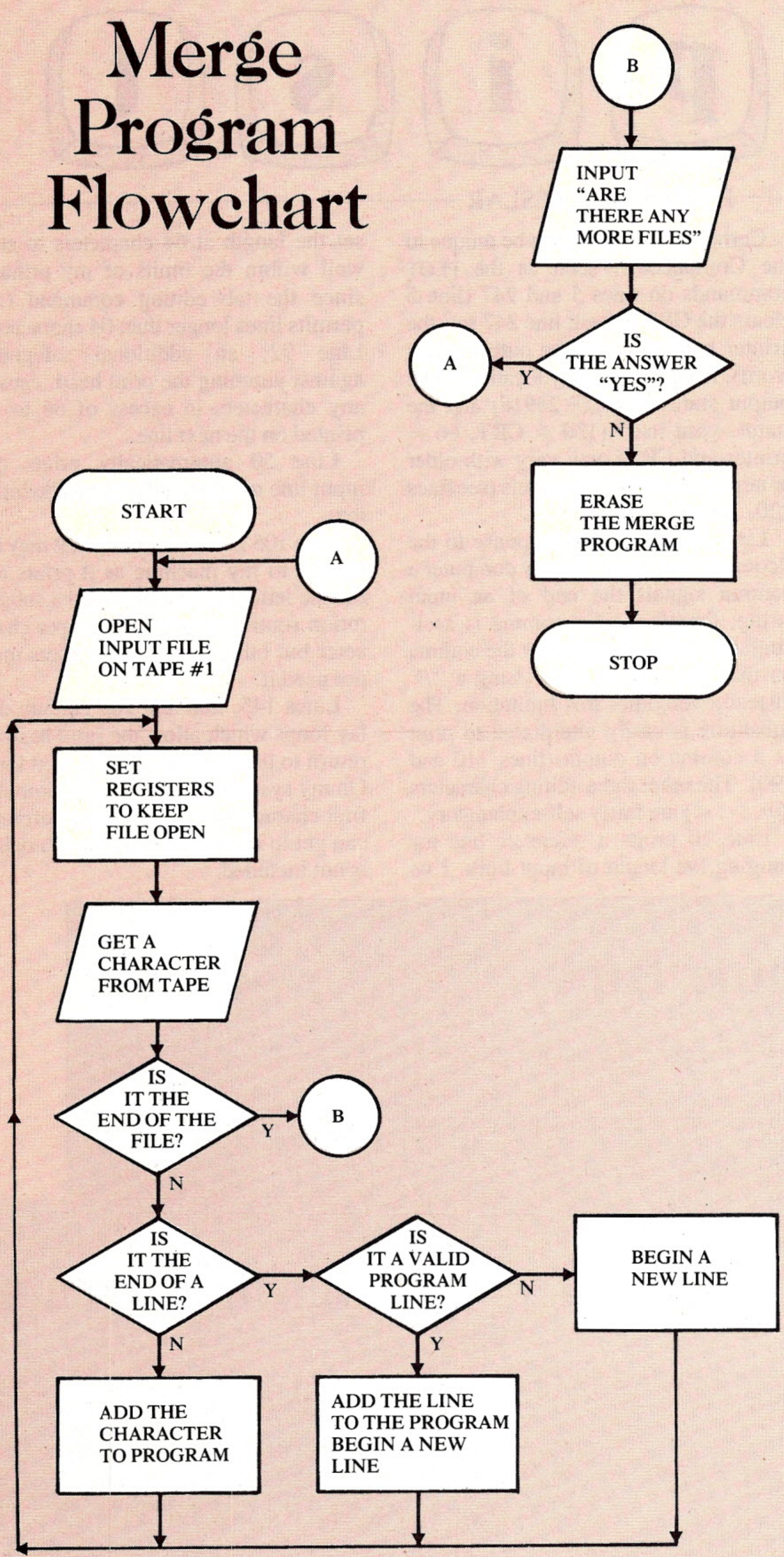
My first problem was to get a listing of the programs like they appear on the screen into a tape file. I asked myself, "If I can list the programs on a printer, why can't I list them on tape the same way?" I opened an output file on tape #1, then told the tape to "listen" by typing "CMD1". Finally I listed the program. Success! Instead of listing the program on the screen, the Pet listed it on tape #1. I closed file 1 to ensure that the cassette buffer was emptied. With testing I found I could list all or any portion of the program depending on what I needed.

At this point, I wrote a first draft of the Merge program. When I ran the program, it opened the file I had created, printed the first line of programming on the screen, and performed a carriage return on that line. The line was now part of the new program.

Program Listing

```
63001 OPEN 1,1,0

63002 POKE 610,1: POKE 578,1: POKE 588,1: POKE 598,0:
        PRINT "Clear Down Down"

63003 GET #1,A$: IF (ST) AND 64 THEN CLOSE 1:
        POKE 525,0: GOTO 63009

63004 IF A$ = CHR$(13) THEN 63006

63005 B$ = B$ + A$: GOTO 63003

63006 IF VAL(B$) = 0 THEN B$ = "": GOTO 63003

63007 PRINT B$;: B$ = "": PRINT: PRINT "RUN 63002":
        PRINT "Home"

63008 POKE 525,3: FOR X = 1 TO 3: POKE 525+X,13:
        NEXT: END

63009 R$ = "RUN 63010": INPUT "ANY MORE FILES"; A$:
        IF LEFT$(A$,1) = "Y" THEN 63001

63010 Y = 0

63011 PRINT "Clear Down Down": PRINT "63010 Y=6":
        FOR X = 1 TO 6: PRINT X+Y+63000: NEXT

63012 POKE 525,9: PRINT R$;"Home": FOR X = 1 TO 9:
        POKE 525+X,13: NEXT: END
```

Merge Program Flowchart



But when the program went back for the next line, "FILE NOT OPEN ERROR" appeared. In stopping the program to empty the keyboard buffer and then having it run again from line 63003, I was closing the file on tape #1. So I dug up some old articles on files and searched for a hint. Line 63002 is what I found. Memory location 620 contains the number of open files. Location 578 contains the device number of the first file opened. Location 588 contains the file reference number of the first file opened. And finally, location 598 contains the mode, either input or output, of the first file opened. Setting these file registers to the correct values each time the file is used keeps the file open.

The last three lines of the program print the Merge program's line numbers on the screen and then do carriage returns on them. This procedure erases the Merge program, leaving only the new program you've created.

To use the program you need one tape to save the Merge program on and a good quality tape for files. Before listing your programs as files be sure the line numbers of the two programs do not overlap. (To avoid overlapping line numbers, use a renumbering program such as the one in *PC*, March 1979, p. 24.)

Now you're ready to merge your programs by following these steps:

1. Load one of the programs to be merged. (Order is not important since the program uses existing line numbers.)
2. Insert the file tape and rewind.
3. Type "OPEN1,1,1:CMD1".
4. List the portion of the program you wish merged.
5. Type "CLOSE1".
6. Load the next program to be merged.
7. Insert the file tape but do not rewind.
8. Type "OPEN1,1,1:CMD1".
9. List the wanted portion of the program.
10. Type "CLOSE1".
11. If you want to merge more programs, go to step 6.
12. Load Merge program.
13. Insert file tape, rewind and run.

Each time the program finishes a file, it will ask if you want to merge any more programs. You can merge as many files as you want. When you answer that there are no more, the Merge part of the program will be destroyed and you'll be left with your new program. □

TV Typist

BY PATRICK SESLAR

Interested in using my microcomputer for word processing, I developed this program to assist in typing letters or other text information.

Although the program provides several time-saving shorthand notations, correcting mistakes must be done on the CRT on an as-typed basis using the backspace and re-entering text where necessary on the current line. Once a line has been entered by hitting the return key, it's too late to make changes on that line. For my purposes, the limited editing ability of the program has presented no problem; I use the program for short letters and for writing descriptions of my other programs for future reference.

This program, written for a Compucolor with 16K RAM, uses an ASR-33 Teletype for output. Only four characters are prohibited: those on lines 15 through 18, since they initiate editing actions later in the program.

Certain commands may be unique to the Compucolor, such as the PLOT commands on lines 5 and 247 (line 5 clears the CRT screen; line 247 sets the printer baud rate and the output status word). Both the memory location of the output status word (−24918) and the status word itself (130 = CRT, 66 = printer and CRT) may vary with older or newer Compucolor models (see lines 220, 247 and 260).

Line 10 reflects my response to the "comma" on input. On my computer a comma signals the end of an input string; therefore, if a comma is accidentally input, all text after the comma on that line will be lost. Using a "/" instead overcomes this limitation. The substitute is easily interpreted to print as a comma on output (lines 110 and 190). The other three editing characters (@, >, <) are fairly self-explanatory.

Line 20 prints a reference line for gauging the length of input lines. I've set the length at 64 characters to stay well within the limits of my printer, since the tab editing command (>) permits lines longer than 64 characters. Line 92, an additional safeguard against jamming the print head, causes any characters in excess of 64 to be printed on the next line.

Line 50 automatically prints the input line number and accepts the input text.

Line 105 is a reaction to what may be unique to my machine as it prints a Ø for the letter O. I've provided a substitution routine to print the proper character but other Compucolor users may not need it.

Lines 145, 200 and 265 contain delay loops which allow the print head to return to the beginning of the next line. On my system, the printer will drop the first character of the next line before it can get to its starting position if a delay is not included.

Sample Run

```
USE '/' FØR CØMMA
USE '@' FØR END ØF LINE
USE '>' FØR TAB 5 SPACES
USE '<' FØR END ØF TEXT
----------------------------------------------------------------
 1  SAMPLE ØUTPUT ØF PRØGRAM
 2  @@@DEAR JØHN/@@>HØW ARE YØU?  I'VE BEEN UP TØ SØME NEW TRICKS
 3  WITH MY CØMPUCØLØR CØMPUTER/ SUCH AS THIS TV LETTER WRITER.
 4  @@>>>>SINCERELY/@@>>>>PAT
 5  <
INPUT CØMPLETE
SAMPLE OUTPUT OF PROGRAM

DEAR JOHN,

     HOW ARE YOU?  I'VE BEEN UP TO SOME NEW TRICKS
WITH MY COMPUCOLOR COMPUTER, SUCH AS THIS TV LETTER WRITER.

                    SINCERELY,

                    PAT

DØ YØU WISH TØ ØUTPUT TØ PRINTER? NO

READY
```

Program Listing

```
5 PLØT 12
7 DIM A$(7,10)
10 PRINT "LETTER PRINTER":PRINT
15 PRINT "USE '/' FØR CØMMA"
16 PRINT "USE '@' FØR END ØF LINE"
17 PRINT "USE '>' FØR TAB 5 SPACES"
18 PRINT "USE '<' FØR END ØF TEXT"
20 FØR X=1 TØ 64: PRINT "-";:NEXT
30 PRINT:CLEAR 4500
```

continued

The program is structured to print out text on the CRT to allow proofreading prior to hard copy printing. Regrettably, there is no correction routine in the program. If an error is discovered it must either be ignored or the program rerun and all text reentered.

If the text passes proofreading, you may go to printer output. Line 245 gives you time to make sure the printer is on and paper positioned. Any input response causes the program to begin printing.

When output to the printer is complete, the program will ask again if output to the printer is desired. YES responses allow for multiple originals while a NO terminates the program.

The most obvious modification to my program would be to develop a correction routine to be inserted after line 225. Such a routine could let you enter a line number followed by the corrected text.

A second relatively simple modification might allow text to be saved on disk for future or frequent use. I have not included this capability in my program, since my Compucolor FCS (File Control System) is different from the file systems I've observed on other microcomputers. □

```
35 FØR Y=0 TØ 6
40 FØR X=1 TØ 10
50 PRINT (10*Y)+X;:INPUT" ";A$(Y,X)
60 IF A$(Y,X)="<" THEN 80
70 NEXT X:NEXT Y:PRINT:PRINT
80 PRINT "INPUT CØMPLETE":I=0
90 FØR Y=0 TØ 6:FØR X=1 TØ 10
91 FØR Z= 1 TØ LEN (A$(Y,X))
92 IF LL>64 GØTØ 265
100 B$ = MID$(A$(Y,X),Z,1)
105 IF B$="Ø" THEN 185
110 IF B$="/" THEN 190
120 IF B$="@" THEN 200
130 IF B$=">" THEN 210
135 IF B$="<" THEN 220
140 LL=LL+1:PRINT B$;
145 NEXTZ:PRINT:LL=0:FØR DD=1 TØ 300:NEXT DD
147 NEXT X:NEXTY
180 GØTØ 220
185 PRINT"0";:LL=LL+1:GØTØ 145
190 PRINT ",";:GØTØ 145
200 PRINT:FØR DD=1 TØ 300:NEXT DD:GØTØ145
210 LL=LL+5:PRINT "     ";:GØTØ 145
220 PØKE-24918,130
225 PRINT:INPUT"DØ YØU WISH TØ ØUTPUT TØ PRINTER? ";D$
230 IF D$="YES" THEN 245
240 GØTØ 270
245 INPUT "IS PRINTER ØN? ";E$
247 PLØT 27:PLØT 146:PLØT 49:PØKE-24918,66
250 GØTØ 90
260 PØKE-24918,130
264 GØTØ 270
265 PRINT:FØR DD=1TØ 300:NEXT DD:LL=0:GØTØ 100
270 END
```

This Johnny Can Run the Computer

BY ELIZABETH A. WHALEN

Johnny can't read? Alphabetize? Syllabicate? This Johnny can!

Johnny finished second grade but couldn't read, add, subtract or do his lessons. He didn't have the necessary skills to function in third grade. To help Johnny with his problems his mother hired me as a private tutor.

At first we worked with pencil and paper, flash cards and books. Then, after my husband bought a Pet, the tutoring changed dramatically! I needed a way to give Johnny fifty problems in addition or subtraction quickly and offer praise for the correct answer. My husband put out a call to that mysterious, invisible network of computer owners and MATH (an uncopyrighted program which he modified to fill my needs) was loaned for us to copy.

MATH is programmed for addition, subtraction, multiplication and division. I can use the operations singly or in pairs. The degree of difficulty of the problem is dictated by the user. When we first started working this program, Johnny needed some basic addition in his memory bank (Johnny likes that term). I would push the right keys and simple problems would appear on the screen. I could control the difficulty with each fifty problems. It wasn't long before I included subtraction. The drill was magnificent! Johnny learned to operate the Pet as well as add and subtract.

His classroom teacher suggested Johnny needed work on alphabetizing. So my husband disappeared into our back bedroom with the Pet and in about two hours he had programmed ALPHA. No bells and whistles are on this program except praise for a correct answer. It is a very functional program any child can run and any adult can program. With the selection of the right key I can produce lists of up to ten words for him to alphabetize. It doesn't take much time because he doesn't need to type the entire word, only the number of the word in the right order.

When Johnny started working with syllabification at school (the second time around) it became clear he was lost. So again, my husband disappeared with the Pet for a couple of hours and program SYBL was created. SYBL lets me drill on breaking up the words into proper syllables. First the word is printed on the screen. Johnny reads it aloud while I listen for the use of new word attack skills. If the word is entirely new we spend time working on its pronunciation. Then the Pet asks how many syllables are in the word. Johnny says the word out loud for me and we talk about vowel sounds and pieces of words. When he has decided just how many syllables are contained in the word he presses the correct number on the keyboard. Next he must divide the word correctly, using the keyboard. Each syllable is followed by hitting the return key until the word is finished. Then the computer tells him whether he is right or wrong. If he has completed the sequence incorrectly it reprints the word and we start over. When he is correct he receives printed praise as well as my positive oral reinforcement.

When Johnny first came to me for tutoring he had no sight vocabulary. We worked with flash cards and reading aloud before the advent of the Pet. I wanted a reading program which would insert his name and make the reading lesson very personal. Again my husband holed up with the Pet for a couple of hours and STORY was on tape. STORY is a list of short sentences which make up a sever line story. It's programmed to print the name of the child and one of his/her friends. Ours are goofy stories because the Pet makes a random selection of the sentences, programmed to produce the shortest story available.

We have incorporated words Johnny has in his memory bank, new words he will need everyday and a few words from science, such as pterodactyl. This is the program he asks to use most frequently. He likes to see his name on the screen and laughs uproariously as strange little stories appear in front of him. Johnny reads aloud for me. He has

Illustration by Josh Randall

ALPHA
Sample Run

```
THIS PROGRAM DRILLS ON ALPHABETIZING A LIST OF WORDS.
THE LIST OF WORDS WILL APPEAR IN RANDOM ORDER AND THE STUDENT
IS TO TYPE THE NUMBERS IN CORRESPONDING ORDER TO PUT THE WORDS
IN ALPHABETICAL ORDER
THE STUDENT'S SCORE WILL BE GIVEN AFTER THE LAST ENTRY.

HIT ANY KEY WHEN READY TO START

ENTER LENGTH OF LIST
MUST BE LESS THAN 10
     ? 5
WHAT IS YOUR NAME
    ?JEFF
WELL JEFF LET'S SEE HOW WELL YOU CAN PUT WORDS IN ORDER
WHEN THE LIST OF WORDS APPEARS ENTER THE NUMBERS IN FRONT OF THE
WORDS IN THE RIGHT ORDER TO ALPHABETIZE THE LIST
HIT RETURN AFTER EACH NUMBER IS ENTERED

PRESS ANY KEY JEFF WHEN YOU ARE READY

           1        MOTOR
           2        THROW
           3        NARROW
           4        WARM
           5        THRASH

TYPE THE NUMBER FOR ALPHABETICAL ORDER
HIT RETURN AFTER EACH NUMBER
IF YOU MAKE A MISTAKE TYPE '10' FOR THE NEXT ENTRY
AND THE PROBLEM WILL START OVER WITH THE SAME LIST

           1
           3
           5
           2
           4

           JEFF LIST                PET LIST

           MOTOR                    MOTOR
           NARROW                   NARROW
           THRASH                   THRASH
           THROW                    THROW
           WARM                     WARM

YOU GOT 5 RIGHT OUT OF 5

SUPERDUPER! !#$&!!

WOULD YOU LIKE ANOTHER LIST (Y-N)
```

SYBL
Sample Run

```
WHAT IS YOUR NAME?

?JEFF

THIS PROGRAM HAS A LIST OF WORDS OF UP TO 3 SYLLABLES

YOUR JOB, JEFF, IS TO TELL HOW MANY SYLLABLES EACH WORD

HAS THEN ENTER THE SYLLABLES IN THE RIGHT ORDER

PRESS ANY KEY

SAT

HOW MANY SYLLABLES?

? 1

RIGHT YOU ARE!!

ENTER THE SYLLABLES AND PRESSRETURN AFTER EACH ONE

? SAT

JEFF YOU ARE BECOMING A GENIUS

LEGEND

HOW MANY SYLLABLES?

?2

RIGHT YOU ARE!!

ENTER THE SYLLABLES AND PRESS RETURN AFTER EACH ONE

? LE

?GEND

SORRY, BUT THAT IS NOT QUITE RIGHT  PLEASE TRY AGAIN
```

even asked to run this program at the end of his lesson while I talk to his mother, making a lesson without a teacher.

Occasionally I use a game which helps teach logic. It is a sophisticated version of tic tac toe called MAXIT.

MAXIT consists of a graphic square filled with small squares, each containing a positive or negative number. It can be played against the Pet or against another person. The object of the game is to get the highest score and force the opponent to take negative numbers. Here the players must think ahead several moves and say "what if". Johnny learns to compute his score and that of his opponent, think ahead to possible moves that he or the Pet can make and work with negative numbers.

The Pet has changed my tutoring style! Drills become games; reading, a quick positive achievement. All of the programs I use are quick to program: A couple of hours produces something I can use comfortably and successfully. The programs can be used in the classroom or in private tutoring.

Educationally, the computer is a giant. It can help a child accomplish goals while enjoying the experience. It can also help teachers, freeing them to work with others while giving children the needed drills.

We've heard that Johnny can't read. Well, this Johnny can read, compute, do his daily lessons *and* run a computer! □

STORY
Sample Run

```
WHAT IS YOUR NAME?  JEFF
WHO IS YOUR FRIEND?  DIANE

JEFF AND DIANE
FLEW TO THE JUNGLE
WHILE THERE
JEFF SAW A HORSE
AND WAS REALLY SCARED
DID YOU SEE THE DRAGON?
IT WAS ORANGE AND PURPLE AND GREEN!

DO YOU WISH ANOTHER VERSION (Y-N)
? Y

JEFF AND DIANE
WENT INTO THE WOODS
WHEN THEY ARRIVED
JEFF RAN TO A LEDGE
AND GIGGLED WITH EXCITEMENT
DID YOU SEE THE UNICORN?
HE HAS A PURPLE NOSE!

DO YOU WISH ANOTHER VERSION (Y-N)
?Y
```

STORY Program Listing

Note: In these listings, "PRINT 3" means "clear the screen".

```
1 REM PROGRAM TO WRITE A STORY
2 REM AUTHOR BOB WHALEN APRIL 1978
10 PRINT "WHAT IS YOUR NAME?"
20 INPUT B$(1)
30 PRINT"WHO IS YOUR FRIEND?"
40 INPUT B$(2)
45 FOR I=1 TO 4:PRINT:NEXT
50 FOR I = 1 TO 5
60 READ A$(I)
70 NEXT I
80 DATA "WENT INTO THE WOODS","WENT TO TOWN","WENT TO THE BEACH"
85 DATA "FLEW TO THE SOUTH PACIFIC","FLEW TO THE JUNGLE"
90 FOR I=1 TO 6
100 READ C$(I)
110 NEXT I
120 DATA "WHILE THERE","WHEN THEY ARRIVED","BEFORE THEY LEFT"
125 DATA "NEXT DAY","THAT EVENING","NEXT MORNING"
130 FOR I=1 TO 6
140 READ D$(I)
150 NEXT I
160 DATA " SAW A HORSE"," RAN TO A LEDGE"," TRIPPED ON A LOG"
165 DATA " SAW A BIG BEAUTIFUL BIRD"," CLIMBED A TREE"," SLID DOWN A BANK"
170 PRINT B$(1);" AND ";B$(2)
180 C=RND(TI)
190 C=INT(C*6)
195 IF C<1 GOTO 180
200 PRINT A$(C)
210 D=RND(TI)
220 D=INT(7*D)
225 IF D<1 GOTO 210
230 PRINT C$(D)
240 E=RND(TI)
250 E=INT(7*E)
255 IF E<1 GOTO 240
256 H=RND(TI)
257 IF H<.5 THEN X$=B$(1)
258 IF H>.5 THEN X$=B$(2)
260 PRINT X$; D$(E)
270 FOR I=1 TO 5
280 READ F$(I)
290 NEXT I
300 DATA "AND WAS REALLY SCARED","AND WAS EXCITED","AND SAID, WOW!"
306 DATA "AND SAID WHAT A THRILL!","AND GIGGLED WITH EXCITEMENT"
310 G=RND(TI)
320 G=INT(6*G)
330 IF G<1 GOTO 310
340 PRINT F$(G)
345 GOSUB 500
350 FOR I=1 TO 3:PRINT:NEXT
360 PRINT"DO YOU WISH ANOTHER VERSION(Y-N)"
370 INPUT X$
380 IF X$="N" GOTO 400
390 GOTO 45
400 STOP
500 FOR I=1 TO 4
510 READ K$(I)
520 NEXT I
530 DATA "DID YOU SEE THE DRAGON?","DID YOU SEE THE PTERODACTYL?"
540 DATA "DID YOU SEE THE GIANT WORM?","DID YOU SEE THE UNICORN?"
550 C1=RND(TI)
560 C1=INT(5*C1)
570 IF C1<1 GOTO 550
580 PRINT K$(C1)
590 FOR I=1 TO 5
600 READ L$(I)
610 NEXT I
620 DATA "HE ROARED IN THE CAVE","HE HAS A PURPLE NOSE!"
```

```
870 PRINT"NOW ";N$;", HOW DID YOU GUESS THAT"
871 PRINT"ANSWER, ITS RIGHT!!"
872 RETURN
900 DATA"FOCUS","FO","CUS","","LEGEND","LEG","END",""
910 DATA"VACANT","VA","CANT","","SPIRAL","SPI","RAL",""
920 DATA"FLAVOR","FLA","VOR","","SOLID","SOL","ID",""
930 DATA"FUTURE","FUT","URE","","MOTOR","MO","TOR",""
940 DATA"PALACE","PAL","ACE","","EVER","EV","ER","","NECK","NECK","",""
950 DATA"NEAR","NEAR","","","NEAT","NEAT","","","PITTER","PIT","TER",""
960 DATA"PICNIC","PIC","NIC","","GARDENER","GAR","DEN","ER","SIX","SIX","",""
970 DATA"GINGERBREAD","GIN","GER","BREAD","GRANDFATHER","GRAND","FA","THER"
980 DATA"TEACHER","TEACH","ER","","TOGETHER","TO","GETH","ER"
990 DATA"TEARS","TEARS","","","SIT","SIT","","","SELL","SELL","",""
1000 DATA"THOUGHT","THOUGHT","","","THROUGH","THROUGH","",""
1010 DATA"TROUBLE","TROU","BLE","","MONEY","MON","EY","","BELL","BELL","",""
1020 DATA"MERRY","MER","RY","","MAYOR","MAY","OR","","FELL","FELL","",""
1030 DATA"POLITE","PO","LITE","","POLICE","PO","LICE","","SURE","SURE","",""
1040 DATA"PLEASE","PLEASE","","","HANDED","HAND","ED","","SAT","SAT","",""
1050 DATA"HANDLE","HAN","DLE","","HAPPEN","HAP","PEN","","BELL","BELL","",""
1060 DATA"WINDOW","WIN","DOW","","WORKMAN","WORK","MAN","","BILL","BILL","","
1070 DATA"WOMAN","WOM","AN","","STANDING","STAND","ING","","SUCH","SUCH","","
```

ALPHA Program Listing

```
1 REM PROGRAM ALPHA
2 REM AUTHOR BOB WHALEN DEC 1978
10 DIM A$(60),B$(60),C$(60),B(60)
15 PRINT"3"
20 PRINT"           *** ALPHABETIZE***
25 PRINT:PRINT
30 PRINT"THIS PROGRAM DRILLS ON ALPHABETIZING A  LIST OF WORDS.
40 PRINT"THE LIST OF WORDS WILL APPEAR IN RANDOM ORDER AND THE STUDENT
50 PRINT"IS TO TYPE THE NUMBERS IN CORRESPONDING ORDER TO PUT THE WORDS
55 PRINT"IN APHABETICAL ORDER
60 PRINT"THE STUDENT'S SCORE WILL BE GIVEN AFTER THE LAST ENTRY.
64 PRINT:PRINT
65 PRINT"HIT ANY KEY WHEN READY TO START
70 GET A$:IF A$="" THEN 70
72 PRINT:PRINT"ENTER LENGTH OF LIST"
73 PRINT"MUST BE LESS THAN 10"
74 INPUT Z:IF Z>9 THEN 73
75 PRINT"3"
80 PRINT:PRINT"WHAT IS YOUR NAME"
90 INPUT N$
100 PRINT"3":PRINT"WELL ";N$;" LETS SEE HOW WELL YOU CAN   PUT WORDS IN ORDER"
110 PRINT"WHEN THE LIST OF WORDS APPEARS ENTER THE NUMBERS IN FRONT OF THE
120 PRINT"WORDS IN THE RIGHT ORDER TO ALPHABETIZE THE LIST
130 PRINT"HIT RETURN AFTER EACH NUMBER IS ENTERED
140 PRINT:PRINT
150 PRINT"PRESS ANY KEY ";N$;" WHEN YOU ARE READY
160 GET A$:IF A$="" THEN 160
180 X=1
200 FOR I=1 TO 58
210 READ A$(I):NEXT
215 RESTORE
220 A=INT(RND(TI)*59)
230 IF A<1 THEN 220
240 FOR J=1 TO Z
250 IF B(J)=A THEN 220
260 NEXT
270 B$(X)=A$(A):C$(X)=A$(A):B(X)=A
280 X=X+1
290 IF X<=Z THEN 220
300 PRINT"3"
310 FOR I=1 TO Z
320 PRINT TAB(3),I,B$(I):NEXT
```

```
630 DATA "IT LOOKED AWFULLY HUNGRY","THE SHARP TEETH WERE SCARY!"
640 DATA "IT WAS ORANGE AND PURPLE AND GREEN!"
650 C2=RND(TI)
660 C2=INT(6*C2)
670 IF C2<1 GOTO 650
680 PRINT L$(C2)
690 RESTORE
700 RETURN
```

SYBL Program Listing

```
10 DIM A$(51),B$(51),C$(51),D$(51)
20 REM      ***SYBL***
30 REM WRITTEN BY R C WHALEN 2/10/79
32 PRINT"3":PRINT:PRINT
35 PRINT:PRINT"WHAT IS YOUR NAME ?
36 INPUT N$
37 PRINT"3":PRINT:PRINT
40 PRINT"THIS PROGRAM HAS A LIST OF WORDS OF UP   TO 3 SYLLABLES
45 PRINT
50 PRINT"YOUR JOB, ";N$;", IS TO TELL HOW MANY   SYLLABLES EACH WORD HAS
55 PRINT
60 PRINT"THEN ENTER THE SYLLABLES IN THE RIGHT   ORDER"
65 PRINT:PRINT"PRESS ANY KEY"
70 GET X$:IFX$=""THEN 70
100 FOR I=1 TO 47
110 READ A$(I),B$(I),C$(I),D$(I)
120 NEXT
130 PRINT"3":PRINT:PRINT
200 A=INT(RND(TI)*48)
210 IF A<1 THEN 200
220 PRINT TAB(9) A$(A)
225 PRINT:PRINT
230 X=1:IF C$(A)<>"" THEN X=2
240 IF D$(A)<>"" THEN X=3
250 PRINT"HOW MANY SYLLABLES ?"
260 INPUT Y:IF Y<>X THEN 700
270 PRINT:PRINT"RIGHT YOU ARE!!"
280 PRINT:PRINT"ENTER THE SYLLABLES AND PRESS RETURN     AFTER EACH ONE"
290 PRINT
300 FOR I=1 TO X
310 INPUT G$(I):NEXT
320 IF G$(1)<>B$(A) THEN 700
325 IF X=1 THEN 800
330 IF G$(2)<>C$(A) THEN 700
335 IF X=2 THEN 800
340 IF G$(3)<>D$(A) THEN 700
350 GOTO800
700 PRINT:PRINT
710 PRINT"SORRY, BUT THAT IS NOT QUITE RIGHT      PLEASE TRY AGAIN"
720 FOR J=1 TO 900:NEXT
730 PRINT"3":PRINT:PRINT TAB(9) A$(A)
740 PRINT:PRINT
750 GOTO 250
800 PRINT"3":PRINT:PRINT
810 GOSUB 840
820 FOR J=1 TO 1900:NEXT
825 PRINT"3":PRINT:PRINT
830 GOTO 200
840 B=INT(RND(TI)*6):IF B<1 THEN 840
845 ON B GOTO 850,855,860,865,870
850 PRINT"SUPER DUPER ";N$
852 RETURN
855 PRINT N$;" YOU ARE BECOMING A GENIOUS"
857 RETURN
860 PRINT"ZOWIE!! ";N$;", A RIPPIN GOOD ANSWER"
862 RETURN
865 PRINT"KAZZOOOOOOOMMM!!, YOU SURE ARE HARD TO  FOOL, ";N$
867 RETURN
```

```
330 B=1:C=1:FOR I=1 TO 10*Z
332 Q=B
334 IF LEN(C$(C))<B THEN B=LEN(C$(C))
336 IF LEN(C$(C+1))<B THEN B=LEN(C$(C+1))
340 IF ASC(MID$(C$(C),1,B))>ASC(MID$(C$(C+1),1,B))THEN GOSUB 700
345 B=Q
350 IF C+2>Z THEN C=0
360 C=C+1:NEXT
370 IF B>5 THEN 390
380 B=B+1:C=1
390 PRINT:PRINT
395 GOSUB 720
400 PRINT"TYPE THE NUMBER FOR ALPHABETICAL ORDER"
410 PRINT"HIT RETURN AFTER EACH NUMBER"
412 PRINT"IF YOU MAKE A MISTAKE TYPE '10'  FOR THE NEXT ENTRY
414 PRINT" AND THE PROBLEM WILL START OVER WITH   THE SAME LIST
420 R=0:T=0:PRINT:PRINT
430 FOR I=1 TO Z
440 INPUT E(I)
442 IF E(I)>Z THEN 300
445 D$(I)=B$(E(I))
450 IF D$(I)=C$(I) THEN R=R+1
460 T=T+1
470 NEXTI
480 PRINT"3":PRINT:PRINT
490 PRINT"          ";N$;" LIST";"   PET LIST "
500 PRINT:PRINT
510 FOR I=1 TO Z
520 PRINT TAB(1),D$(I),C$(I):NEXT
530 PRINT:PRINT
540 PRINT"YOU GOT ";R;" RIGHT OUT OF ";T
542 IF R=T THEN GOSUB 900
550 PRINT:PRINT
560 PRINT"WOULD YOU LIKE ANOTHER LIST (Y-N)"
570 INPUT L$:IF L$="N" THEN 800
580 GOTO 180
600 DATA"FOCUS","LEGEND","VACANT","SPIRAL","CABIN","FLAVOR","SOLID"
610 DATA"FUTURE","MOTOR","PALACE","NARROW","BIKE","LAMP","GIFT"
620 DATA"STAMP","EVER","CHAIR","OVER","BOX","BEND","BUS","BANK","PINCH","POND"
630 DATA"PRETTY","PLAN","MASTER","GHOST","INSIDE","ISLAND","WARM","PATCH"
640 DATA"MATCH","WRONG","SHOP","SHEEP","SHALL","SHIP","GRAND","GRUMPY","GROW"
650 DATA"GREAT","THREAD","THRASH","THROW","THRIFT","THREE","THINK","THROW"
660 DATA"THAN","YARD","WORD","YEAR","WOULD","VERY","YELLOW","MOM","GNU"
700 X$=C$(C):C$(C)=C$(C+1):C$(C+1)=X$
710 RETURN
720 FOR J=1TO2:FOR F=1 TO Z-1
730 IF ASC(C$(F))=ASC(C$(F+1)) THEN GOSUB 1200
740 NEXT F:NEXT J
750 RETURN
800 STOP
900 A=INT(RND(TI)*7)
910 IF A<1 THEN 900
920 M$(1)="EXCELLENT SCORE!!"
925 M$(2)="SUPERDUPER!!#$&!!"
930 M$(3)="GEEWHIZGOLLYSAKE-GOOOD!#!#!"
940 M$(4)="DONE LIKE A GENIUS!!"
950 M$(5)="TRREEEMMENDOUUS!!!!"
960 M$(6)="LOLAPALUZA SCORE!!!!"
970 PRINT:PRINT:PRINT M$(A)
980 RETURN
1200 G$(F)=C$(F):G$(F+1)=C$(F+1)
1210 X=2
1220 IF ASC(MID$(C$(F),X,1))<ASC(MID$(C$(F+1),X,1))THEN 1300
1230 IF ASC(MID$(C$(F+1),X,1))<ASC(MID$(C$(F),X,1))THEN 1290
1240 X=X+1
1250 IFX>LEN(C$(F)) THEN 1300
1252 IF X>LEN(C$(F+1)) THEN 1300
1260 GOTO 1220
1290 C$(F+1)=G$(F): C$(F)=G$(F+1)
1300 RETURN
```

How to Build a Program

BY L. MITCHELL WEIN

Well, the TRS-80's home and you've read the manual. You've also played a few computer games. Now what?

Now comes the big challenge: writing your own programs and controlling that burgeoning tape library.

Before starting to program, however, it's essential that you insulate your 16K Level II machine against keyboard bounce which can work havoc with the editing features used to correct errors. The best way to prevent keyboard bounce is to enter one of two available machine language programs right after turning on your machine: KBFIX, produced by Radio Shack, or Microsoft's Level III. KBFIX leaves you 15515 bytes free while Level III leaves 10493 bytes to start. So, unless you wish to use Level III's other special features (mainly abbreviations, graphics and line renumbering), you should use KBFIX.

Now you're ready to begin a program! It's best to proceed from the known to the unknown with standard locations for your program segments. Use lines 1 to 19 for a statement of the problem to be solved, displayed and/or stored; include your name, address and the date. Revise the date each time you add to the program.

On lines 20 to 299 enter in standard English all of the detailed steps to resolve your problem. Enter each step at line intervals of 10, writing in between only if forced to do so. Use an apostrophe (') or REM after every line number from 1 to 1999 to indicate the lines are for your own use — not the computer's.

The detailed steps are your logic statements. Include everything you can think of. Always use the tab control (right arrow) to keep your starting positions lined up and the line feed control (down arrow) at the end of each line. Otherwise, if you attempt to list your program on a printer you may lose some of your work.

A typical sequence of logic statements which can be used in many situations is:

```
20'  PRINT HEADER
30'  INITIALIZE VARIABLES
40'  DISPLAY MENU
50'  INPUT DATA FROM TAPE
60'  INPUT DATA FROM KEYBOARD
70'  DISPLAY TABLE
80'  OUTPUT DATA TO TAPE
```

On lines 300 to 999 list your variable definitions. Again, be as complete as you can. If something occurs to you later, be sure to add it.

```
300'  VARIABLE DEFINITIONS:
310'  ML=MENU LOCATOR
320'  PT$=PAGE TURNER
330'  LN=LINE NUMBER
```

On lines 1000 to 1999, list subroutine locations, which should be higher than 100 times the highest logic statement but lower than 30,000. (The reasons for this will become clear later.) A typical sequence of essential subroutine locations:

```
1000'  SUBROUTINE LOCATIONS:
1010'  13000 CLR SCREEN AND CENTER CURSOR-
       LARGE PRINT
1020'  14000 CLR SCREEN AND CENTER CURSOR-
       SMALL PRINT
1030'  15000 TURN PAGE-LARGE PRINT
1040'  16000 TURN PAGE-SMALL PRINT
1050'  17000 PAGE HEADER FOR TABLE
1060'  18000 ARRAY CALCULATIONS
```

When you start to write a program in BASIC, begin each segment at line locations which are 100 times the related logic statement's location. This way, you can easily flip back and forth between logic statements and program statements as required. It would probably help if you printed or otherwise copied logic statements, variable definitions and subroutine locations before starting to write program statements.

Each subroutine should have an END statement before it. Program statements for the subroutines indicated in lines 1010, 1020, 1030 and 1040 are:

```
12999  END
13000  CLS:PRINTCHR$(23):PRINT:PRINT:PRINT:
       PRINT:PRINT:RETURN
13999  END
14000  CLS:PRINT:PRINT:PRINT:PRINT:
       PRINT:RETURN
14999  END
15000  PRINT@972,"PRESS ENTER TO CON-
       TINUE";:INPUTPT$:CLS:PRINTCHR$(23):
       LN=0:RETURN
15999  END
16000  PRINT@998,"PRESS ENTER TO CON-
       TINUE";:INPUTPT$:CLS:LN=0:RETURN
```

Notice that the line-feed control (down arrow) was used for lines over 64 characters with text positioned to allow easy viewing of the line numbers.

After laying out your problem, logic statements, variable definitions and subroutine locations by the use of REM (') statements on lines 1 to 1999, you're finally ready to write the actual program.

To begin, let's construct the header indicated in line 20 of our logic statements.

```
2000  GOSUB 13000: PRINT TAB(6) "GREAT PRO-
      GRAM":PRINT:PRINT TAB(2) "BY JOHN
      SMITH"
2010  GOSUB 15000
```

The two-line module prints the header in the center of the screen using a 32 characters per line format and holds the header page until Enter is pressed.

Following the header, we initialize variables using some or all of the following steps:

```
3000  CLEAR 500     (reserve bytes for string usage)
3010  DEFINT A,B,C    (or DEFDBL and/or
      DEFSTR)
3020  GOSUB 13000: PRINT"ENTER MAXIMUM":
      PRINT"NUMBER OF ROWS"
3030  INPUT "FOR TABLE": TB
3040  DIM XY(TB,8)
3050  A=2: B=20: C=100
3060  CLS: PRINT CHR$ (23)
```

The order in which CLEAR, DEFINT (and/or DEFDBL/DEFSTR) and DIM are presented is important. Any variation of this order could cancel one or both of the latter statements. Lines 3020 and 3030 determine the size of the array (table) while line 3040 establishes eight columns for the table. Line 3060 turns the page and establishes 32 character/line format.

The next module of our program is the MENU, which is the most important part of any program as it allows the user to survey all options and make a choice. In addition, using the MENU approach allows you to construct separate program modules as indicated by the logic statements. Lines 4070, 4090 and 4100 are an error trapping routine.

```
4000  PRINT TAB(11)"MENU":PRINT
4010  PRINT"1. INPUT DATA FROM TAPE."
4020  PRINT"2. INPUT DATA FROM
       KEYBOARD."
4030  PRINT"3. DISPLAY TABLE."
4040  PRINT"4. OUTPUT DATA TO TAPE."
4050  PRINT"5. END PROGRAM."
4060  PRINT:PRINT:PRINT:PRINT:PRINT:PRINT:
      PRINT TAB(10) "ENTER YOUR CHOICE.";:
      INPUT M%
4070  IF M%<1 OR M%>5 THEN 4090
4080  ON M% GOTO 5000, 6000, 7000, 8000, 9000
4090  GOSUB 13000: PRINT"MENU SELECTION
      MUST BE BETWEEN 1 AND 5."
4100  GOSUB 15000:GOTO 4000
```

Inputting a 1 makes the program branch to line 5000 to input data from tape. The input module could be set up as follows:

```
5000  CLS:PRINT CHR$(23)
5010  PRINT TAB(5)"INPUT DATA FROM
      TAPE":PRINT:PRINT
5020  PRINT"LOAD DATATAPE.":
      PRINT:PRINT"PRESS ENTER ON KEY-
      BOARD.":PRINT:PRINT"PRESS PLAY ON
      RECORDER.";:INPUT P$
5030  FOR L%=1 TO TB
5040        FOR S%=1 TO 4
5050              INPUT#-1,XY(L%,S%)
5060        NEXT S%
5065        IF XY(L%,1)=0 THEN 5080
5070  NEXT L%
5080  GOSUB 18000
5090  CLS:PRINT CHR$(23):GOTO 4000
```

Outputting is performed by a subroutine similar in structure to the input routine.

```
8000  CLS:PRINT CHR$(23)
8010  PRINT TAB(5)"OUTPUT DATA TO
      TAPE":PRINT:PRINT
8020  PRINT"LOAD DATATAPE."
      :PRINT:PRINT"PRESS RECORD AND
      PLAY":PRINT"ON RECORDER.":PRINT:
      INPUT"PRESS ENTER ON KEYBOARD.";P$
8030  FOR L%=1 TO TB
8040        FOR S%=1 TO 4
8050              PRINT#-1,XY(L%,S%)
8060        NEXT S%
8065        IF XY(L%,1)=0 THEN 8090
8070  NEXT L%
8090  CLS:PRINT CHR$(23):GOTO 4000
```

The input and output loops from 5030 to 5070 and 8030 to 8070 are exactly alike except for the INPUT# and PRINT# instructions. The similarity is important for data to be transferred successfully to and from tape. In addition, lines 5065 and 8065 allow exit from the transfer loops when the data is exhausted. In this example the transfer loops only affect data in columns 1

to 4 of the table. Line 5080 accesses a subroutine which would calculate and generate columns 5 to 8 of the table. It is important that generated data *not* be transferred to tape in order to maximize tape usage. The keyboard input module would also access the same array calculation subroutine.

Having written a program and transferred in to cassette tape, we recognize that the program will be useless if we don't know where to look for it in the future. Therefore, effective control of tapes also requires logical steps be taken.

When dumping programs to tape, use CSAVECHR$-(13); this allows use of CLOAD without an operand when reloading. Use CLOAD? to verify each program after saving it. If BAD is printed on the display and the CTR-80 stops, you have a tape glitch. There's no need to discard the tape, however; merely note the turn location of the glitch, transfer all other good programs to other tapes and use a bulk eraser on the defective tape. Future programs can then be recorded five turns above and/or below the glitch.

Programs should be recorded at least twice and kept in a minimum of two locations. One location could be an open container attached to a wall for easy access; but the other should be a closed metal box tucked away from disturbance in a drawer. Both locations should be at a safe distance from the computer power supply, video display and television sets.

Keep the tape recorder on the *opposite* side of the keyboard from the power supply and video display. The recorder should be cleaned every second day with "Head Cleaner and Demagnetizer" and every second week with "Tape Recorder Cleaner and Lubricant", both by Realistic.

When you purchase a new cassette, number it and mark the sides A and B. Test each tape for number of turns, then rewind and bulk erase.

An essential part of tape control is maintaining an *accurate* program location log. Paper suitable for this log is 4 column accounting paper, which also has columns for dates and descriptions. Use two columns for each cassette, labelled side A or side B. Above the labels should be the cassette identification data: container #; cassette #; cassette turns. Each program line entry should contain date; description/record character (Level — I, II, III; machine language — ML; data — D); scratch status (X=scratched-erased); starting turn location and ending turn location. □

Date	**1/Cassette#7(111)**		**2/Cassette#8(198)**	
1979 Descrip./Char.	**A**	**B**	**A**	**B**
4/14 Pge Separator/II	005-011			
4/30 Change of D1r/I	X 005-020			
5/10 Marks/D	016-031			

Editors Note: Microsoft Level III is available from local computer stores or may be purchased for $49.95 directly from Microsoft Consumer Products, 10800 N.E. Eighth, Suite 819, Bellevue, WA 98004; (206) 454-1315. KBFIX is available at Radio Shack Stores.

TRS-80 Line Renumbering

BY BLAKE WARD

I was writing one day when I found that unless I changed some line numbers I needed a line numbered 88.5. Since I'm too cheap to buy Radio Shack's renumbering program, I sat down and wrote one myself.

The program, shown in Program Listing 1, is executed by typing "RUN 60000". It asks for the starting line and the interval between each line, and proceeds to renumber your program, changing line numbers and all the GOTOs, GOSUBs and THENs. Renumbering can take a long time (sometimes more than one-half hour), depending on the length of your program. The reason will become apparent as I explain how the program works.

When the program states ENTER STARTING LINE, INCREMENT, enter the new starting line number and the increment size, separated by a comma. The program will check to see if the numbers entered would produce line numbers greater than 60000 (which would be fatal for your program). Conflicting line numbers result in the program requesting input again. If this should happen, either one or both of the input numbers should be decreased.

After the line numbers have been altered you will see LINES CHANGED NOW CORRECTING GOTO, GOSUB, ETC. while the program makes the changes. You might as well get a cup of coffee or watch TV because if your program is long it will take a while. When the program prints FINISHED, you must type DELETE 60000-60350

Program Listing 1

```
6ØØØØ AA=PEEK(16548)+PEEK(16549)*256: AB=AA: AC=AA: V=Ø: REM  LINE
      RENUMBERING PROGRAM BY BLAKE WARD, OXBOW, SASK. CANADA:  TO EXECUTE
      TYPE 'RUN 6ØØØØ' NOTE:  DON'T REMOVE THIS REM STATEMENT....
6ØØ1Ø AC=PEEK(AC)+PEEK(AC+1)*256: V=V+1: IF PEEK(AC+2)+PEEK(AC+3)*256<>6ØØØØ
      THEN 6ØØ1Ø ELSE DIM LL(V): Y=Ø
6ØØ2Ø INPUT "ENTER STARTING LINE, INCREMENT";L1,L2: L3=L1: IF L1+L2*(V-1)=>6ØØØØ
      THEN 6ØØ2Ø
6ØØ3Ø FOR X= Ø TO 3: A(X)=PEEK(AA+X): NEXT: N=A(3)*256+A(2): IF N=6ØØØØ
      THEN 6ØØ6Ø
6ØØ4Ø LL(Y)=N: Y=Y+1
6ØØ5Ø R=INT(L1/256): RR=L1-R*256: L1=L1+L2: POKE AA+2,RR: POKE AA+3,R:AA=
      A(Ø)+A(1)*256: GOTO 6ØØ3Ø
6ØØ6Ø PRINT "LINES CHANGED, NOW CORRECTING GOTO, GOSUB, ETC.": AA=AB-1: Y=Y-1:
      H=PEEK(A(Ø)+A(1)*256)+PEEK(A(Ø)+A(1)*256+1)*256: POKE 16548,H-INT(H/256)*
      256: POKE 16549, INT(H/256)
6ØØ7Ø AA=AA+4: IF PEEK(AA)*256+PEEK(AA-1)=6ØØØØ THEN PRINT "FINISHED":
      POKE 16548,AB-INT(AB/256)*256: POKE 16549,INT(AB/256): END
6ØØ8Ø AA=AA+1: C=PEEK(AA): IF C=Ø THEN 6ØØ7Ø ELSE IF C<>141 AND C<>145 AND
      C<>149 AND C<>159 AND C<>2Ø2 THEN 6ØØ8Ø
6ØØ9Ø AZ=AA
6Ø1ØØ AA=AA+1: C=PEEK(AA): IF C=32 THEN 6Ø1ØØ
6Ø11Ø IF C>47 AND C<58 THEN GOSUB 6Ø14Ø: GOTO 6Ø1ØØ
6Ø12Ø IF LN$="" THEN 6ØØ8Ø ELSE IF C=44 THEN GOSUB 6Ø15Ø: GOTO 6ØØ9Ø
6Ø13Ø GOSUB 6Ø15Ø: IF C=Ø GOTO 6ØØ7Ø ELSE 6ØØ8Ø
6Ø14Ø LN=C-48: LN$=LN$+RIGHT$(STR$(LN),1): RETURN
6Ø15Ø Z=-1: LN=VAL(LN$): FOR X=Ø TO Y: IF LL(X)=LN THEN Z=X: X=Y
6Ø16Ø NEXT: IF Z=-1 THEN PRINT "COULDN'T FIND";LN;"IN TABLE": GOTO 6Ø23Ø
6Ø17Ø NL=L2*Z+L3: NL$=STR$(NL): NL$=RIGHT$(NL$,LEN(NL$)-1)
6Ø18Ø IF LEN(NL$)>AA-AZ-1 THEN 6Ø24Ø
6Ø19Ø IF LEN(NL$)<AA-AZ-1 THEN NL$=NL$+"_": GOTO 6Ø19Ø
6Ø2ØØ FOR X=AZ+1 TO AA-1: LN$=MID$(NL$,X-AZ,1): LN=VAL(LN$)+48: POKE X,LN
6Ø21Ø IF LN$=" " THEN POKE X,32
```

Continued

```
6Ø22Ø NEXT
6Ø23Ø LN$="": RETURN
6Ø24Ø M=AA: AA=AA+1: B=PEEK(M): IF B=Ø THEN B=32: GOTO 6Ø27Ø
6Ø25Ø B1=PEEK(M+1): IF B1=Ø THEN 6Ø28Ø
6Ø26Ø POKE M+1,B: B=B1: M=M+1: GOTO 6Ø25Ø
6Ø27Ø M=M-1
6Ø28Ø J=PEEK(M+3)*256+PEEK(M+2): B2=PEEK(M+4): B3=PEEK(M+5): IF B3*256+
      B2<>6ØØØØ THEN J=J+1
6Ø29Ø POKE M+1,B: POKE M+2,Ø: POKE M+3,J-INT(J/256)*256: POKE M+4,INT(J/256):
      B=PEEK(M+6): POKE M+5,B2: POKE M+6,B3: M=M+6: IF B3*256+B2<>6ØØØØ
      THEN 6Ø25Ø
6Ø3ØØ M=AA
6Ø31Ø IF PEEK(M)<>Ø THEN M=M+1: GOTO 60310
6Ø32Ø W=PEEK(M+4)*256+PEEK(M+3)-L2: M=AA: IF W=5999Ø THEN W=L3+L2*(V-1)
6Ø33Ø IF PEEK(M)*256+PEEK(M-1)<>W THEN M=M-1: GOTO 6Ø33Ø
6Ø34Ø W=PEEK(M-2)*256+PEEK(M-3)+1: POKE M-2,INT(W/256): POKE M-3,W-INT(W/256)*
      256: GOTO 6Ø18Ø
6Ø35Ø REM COPYRIGHT 1979 BY BLAKE WARD  ALL RIGHTS RESERVED
```

Program Listing 2

```
1Ø CLS: PRINT PEEK(16548);"__";PEEK(16549): B=PEEK(16548)+PEEK(16549)*256
2Ø D=B: B=PEEK(D+1)*256+PEEK(D): IF B>Ø GOTO 2Ø
3Ø POKE 16548,D-INT(D/256)*256: POKE 16549,INT(D/256): END
```

(Enter), which erases the renumber program itself. You can then do whatever you want with your program, such as save it on tape or finish writing it.

The biggest disadvantage of writing a renumbering program in BASIC is that you must either load it before you type in your program, or you must type it in each time you wish to use it. Not so with this program! Program Listing 2 contains a short program to allow merging the renumbering program from tape with the program in memory. Type in Listing 2, changing the line numbers to whatever is convenient. Run this short program and make a note of the two numbers which appear on the screen. Then CLOAD the renumbering program as usual. When the renumbering program has finished loading, type POKE 16548,XX: POKE 16549,YY where XX is the first number that was displayed on the screen and YY the second number. You will now have both programs in memory just as if you had typed them in.

To understand how the program works, let's see how Radio Shack's Level II stores BASIC language programs in memory.

Under normal circumstances, a program is stored in memory starting at location 17129 (or 42E9 hexadecimal). Each line is stored in the following manner: The first two bytes are the address of the beginning of the next line of the program, allowing BASIC to "hop" through memory without scanning each line's text. Next are two bytes which represent the current line number in hexadecimal: the first byte plus the second byte times 256 equals the line number in decimal form. Following the first four bytes is the actual line text stored in ASCII (A=65, B=66, etc.). All BASIC keywords are stored in a special, compressed format to save memory (GOTO=141, GOSUB=145, PRINT=178). At the end of the text is a 00 which signifies end of line. The end of the program is represented by three 00s, in other words, the address of the next line of program is 0000 (there isn't one!).

The first line of the renumbering program finds where the program begins (usually 17129). Line 60010 counts the number of lines in your program and DIMensions the array (LL) to that amount. Line 60020 asks for the starting line and increment and checks to see if they're too large. Lines 60030 through 60050 scan your program, placing the old line number in the array (LL) and replacing it with the new line number (L1). It adds the increment (L2) to L1 and hops to the next line. Line 60030 checks to see if it has reached the end of your program; if it has, the program jumps to line 60060. You can't renumber the renumbering program.

Radio Shack's BASIC executes GOTOs and GOSUBs by starting at the beginning of the program and hopping from line to line until it reaches the proper line. In line 60060, we play a trick on BASIC by changing memory locations 16548 and 16549, which contains the starting address of the program. The ruse is necessary because the next part of the renumbering process shifts your program ahead in memory and if BASIC tries to hop through your program to get to the renumbering program, it can become confused and forget most of your program and all of the renumbering program. To avoid catastrophe we make BASIC think the only program in memory starts at line 60000. Line 60070 puts the locations back once the changes have been made and it's safe to hop through your program again.

Line 60080 checks each byte to see if that byte is one of the BASIC keywords such as GOTO or GOSUB; if not, 1 is added to AA (which keeps track of the current memory location) and the next byte is checked.

Lines 60100 to 60230 load the old line number into LN$, look it up in the array and put the new line number in NL$. Line 60180 checks to see if the new line number has more digits than the old number. If it does, the program branches to line 60240.

Lines 60240 to 60340 move the remainder of the program ahead in memory one space to make room for the new line number. Each time part of the program is moved ahead one space, it is allowed to overlap onto the first line of the renumbering program which isn't needed anymore. For this reason the renumbering program must be loaded each time you want to use it. If you remember, at the start of each line of program, the address of the next line is stored. Since all the lines following the line number being corrected have been moved ahead in memory one space, these addresses must increase by one. After shifting the program ahead in memory and correcting the line addresses, the program jumps back to line 60180 to see if there is enough room.

Line 60200 puts the new line number into our program. Each time the end of a line is reached, line 60070 adds four to AA to jump over the four bytes of

information at the start of a line. Line 60070 also checks to see if the next line is 60000 which would mean all of your program has been renumbered. If this is the case, the line corrects memory locations 16548 and 16549 and the program ends.

The part of the renumbering program that takes most of the time is the section that shifts the program ahead in memory (lines 60240 to 60340). One way the program could be made faster would be to write it in assembly language, or to call a machine language subroutine (using USR) to do the slowest part of the program. A machine language subroutine would take a fraction of the time used by BASIC.

Short of rewriting the program in assembly language, there is a change that could make it easier to use. After the program has been running for fifteen minutes or so, you can't help wondering whether it is working right or if it is in an endless loop. One way to tell if the program is working would be to display the number of the line as it is completed.

To display line numbers, add the following line to the program:

```
60075 CLS: PRINT @ 512, "CORRECTION IS COMPLETED FOR";P;"OUT OF";V-1;"LINES":P=P+1.
```

The new line will cause the program to display the line just completed and the total number of lines in your program.

When you type in the renumbering program it is a good idea to store it on tape or diskette before using it because a typing mistake could cause the renumbering program to erase itself. Another suggestion is to use it on a short program first because typing errors could cause your original program to be erased. When I tested the program after typing it, I stored a copy on tape and typed in a few lines like: 3 GOTO 3, 8 ON X GOSUB 3,8,47 and 47 IF X=0 THEN 3. The test lines don't have to be part of a program but they must be correct. You can test the program without wasting much time or losing a program you spent hours typing in!

The renumbering program works equally well in Disk Basic. I have a copy on diskette in ASCII format so that I can use the Merge command to add it to another program in a matter of seconds.

I hope this program adds to the enjoyment you'll get from computer. At the very least, it should make things a little easier. □

A Beautiful Way To Interface

IQ 140

SOROC's first and foremost concern, to design outstanding remote video displays, has resulted in the development of the IQ 140. This unit reflects exquisite appearance and performance capabilities unequaled by others on the market.

With the IQ 140, the operator is given full command over data being processed by means of a wide variety of edit, video, and mode control keys, etc.

The detachable keyboard, with its complement of 117 keys, is logically arranged into 6 sections plus main keyboard to aid in the overall convenience of operation. For example, a group of 8 keys for cursor control / 14 keys accommodate numeric entry / 16 special function keys allow access to 32 pre-programmed commands / 8 keys make up the extensive edit and clear section / 8 keys for video set up and mode control / and 8 keys control message and print.

Two Polling options available: 1) Polling compatible with Lear Siegler's ADM-2. 2) Polling discipline compatible with Burroughs.

IQ 120

The SOROC IQ 120 is the result of an industry-wide demand for a capable remote video display terminal which provides a multiple of features at a low affordable price. The IQ 120 terminal is a simple self-contained, operator / computer unit.

The IQ 120 offers such features as: 1920 character screen memory, lower case, RS232C extension, switch selectable transmission rates from 75 to 19,200 bps, cursor control, addressable cursor, erase functions and protect mode. Expansion options presently available are: block mode and hard copy capability with printer interface. The IQ 120 terminal incorporates a 12-inch, CRT formatted to display 24 lines with 80 characters per line.

SOROC TECHNOLOGY, INC.

165 FREEDOM AVE., ANAHEIM, CALIF. 92801

(714) 992-2860 / (800) 854-0147

STONEHENGE

BY BONNIE J. AND DAVID J. BEARD

Stonehenge — a mysterious structure of massive stones on Salisbury Plain in England — now stands sadly dilapidated after four millennia of weather, war and vandalism. The question of who built it and why has generated a lot of interest and controversy over the centuries. More than one person guessed that Stonehenge was used as temple-observatory. But recent studies, made possible by modern computers, suggest that it is a much more sophisticated machine than anyone had ever supposed.

Many of the functions embodied in the design of Stonehenge lend themselves neatly to implementation on a small computer. Experimenting with these functions can provide valuable insights into astronomy, programming techniques and Stonehenge itself.

Stonehenge is shaped something like a large skillet (Figure 1), outlined by chalk banks. The circular body, 100 meters in diameter, encloses the well-known standing stones. The "handle" is an avenue pointing to the northeast in the direction of midsummer sunrise. Within the bank is a circle of 56 evenly spaced Aubrey holes, named for John Aubrey, who reviewed Stonehenge for his king in the 1660s. These holes have been refilled with chalk to serve as markers or calibrations. Inside the ring of Aubrey holes is a circle of 30 sarsen stones capped with lintels, and in the center of the structure are the five great trilithons arranged in a horseshoe.

Many people have speculated about who built Stonehenge and why. Some thought Merlin transported the stones from Ireland and erected the monument to British dead. Others thought it might have been a temple built by the Romans, or a royal court of the Danes, or a Druid temple. In the 1740s, Dr. William Stukeley was the first writer to mention the avenue and the fact that the entire structure lines up to the northeast and the midsummer sunrise. For the last five hundred years, the Druids have been the most popular; people liked to envision bloody-handed priests making human sacrifices and chanting mystical nonsense to the Moon.

Today's best guess is that Stonehenge was started around 2000 B.C. and built over a period of three centuries, with many modifications and additions. It was first laid out by a neolithic agricultural people, primarily as an astronomical observatory used to synchronize their calendar with the seasons. Modern man takes an accurate calendar for granted, but determining the time of year was a difficult problem in ancient England, where fog and rain obscured the sky for months at a time. It was also a matter of vital importance; a village lured into planting before the last of the deep frosts might literally starve the next winter.

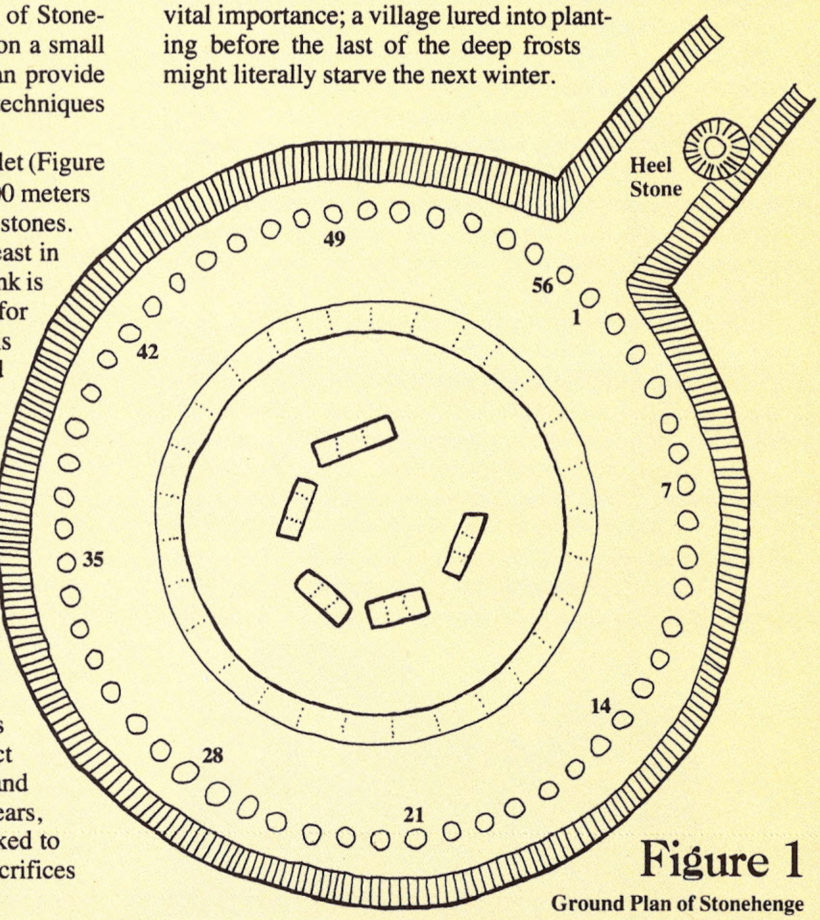


Figure 1

Ground Plan of Stonehenge

The location of Stonehenge is significant. The Salisbury Plain was considered holy long before Stonehenge was erected. There are an unusually large number of barrow burials in this area and artifacts found there suggest that the plain was also a site for trade fairs and religious festivals. There was probably a resident priesthood there before Stonehenge was built. History and literature hint that these priests were not only religious leaders but also teachers, doctors and respected authorities on many practical subjects. In fact, the nearest equivalent today to Stonehenge might well be a modern state agricultural college, complete with fairs, festivals and football games.

Many archaeologists doubted that Stonehenge was ever used as an observatory, arguing that primitive people would not have so sophisticated a knowledge of astronomy. Most of these doubts were dispelled in 1963 by Gerald Hawkins. Using an IBM 7090 computer at Harvard College Observatory, Hawkins systematically searched for alignments between key stations at Stonehenge and significant Sun and Moon positions. He expected to find some correlation, but he was astonished at the number and accuracy of the alignments he found.

The use of Stonehenge as a giant astrolabe is beyond the scope of this article, but it should be noted that many unrelated sighting lines were combined into a simple, elegant, symmetrical structure — an intellectual feat of no mean order.

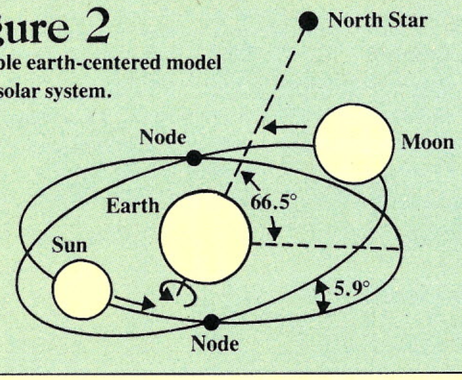
Soon afterward, teased by historical references to something significant that happened at Stonehenge every 18 or 19 years, Hawkins wondered if the builders of Stonehenge might have been predicting eclipses. As we will see later, 56 is a very important number to anyone interested in eclipses. There were exactly 56 Aubrey holes, and it is not easy or convenient to divide a circle into 56 equal sectors. He also noted the correspondence between the 29.5 day lunar month and the 30 stones and 29 arches of the sarsen circle.

Machine analysis had shown that Stonehenge was an accurate astronomical instrument. Hawkins wondered if it was also a sort of neolithic computer, where men used integer mathematics to track the wanderings of the Sun and Moon. A decade later Fred Hoyle elaborated on Hawkin's work, suggesting a simpler and more comprehensive method of operating the Stonehenge computer. The model and method of operation presented here are borrowed primarily from Hoyle's book.

Figure 2 shows a simple Earth-centered model of the universe as seen from somewhere above the northern hemisphere. The Earth is rotating counter-clockwise once each day. Its axis of rotation is always tilted towards the North Star. The Sun and Moon travel counter-clockwise around circular paths, the Sun taking one year for a full circuit, the Moon taking 27.32 days. The Moon's orbit is tilted slightly with respect to the Sun's. The points where the Sun's and Moon's orbits cross are the *nodes* of the Moon. These nodes are not stationary but drift slowly clockwise, making a full circuit in 18-2/3 years, or 3 circuits in 56 years. Beyond the

Figure 2

A simple earth-centered model of the solar system.



Sun and Moon are the immovable stars, providing a fixed reference for measuring the various movements.

Because the Earth is rotating much more quickly than the Sun and Moon, an observer in the Northern Hemisphere will see the Sun and Moon rise on his left, travel clockwise across the sky, and set on his right. However, each day the Sun and Moon will have drifted slightly counter-clockwise against the background of stars. Although very simple, this model of the cosmos can explain and predict the seasons, the phases of the Moon, the time of moon rise, and the patterns of the ocean's tides and the occurrence of solar and lunar eclipses! If Stonehenge was used to implement this model, we can understand why so much construction effort was justified.

Look at Figure 2 again. When the Sun lies in the direction of the North Star, the Northern Hemisphere is tilted toward the Sun. The Sun will appear high in the sky and it will be summer. When the Sun lies opposite the North Star, appearing low in the sky, it will be winter. When the Sun stands highest in the sky, it is the summer *solstice,* or midsummer's day (solstice is simply Latin for "the sun stands").

When the Sun and Moon are on opposite sides of the Earth, the entire earthward side of the Moon is lighted, and it is full moon. The Moon will rise and set exactly twelve hours behind the Sun and stand directly overhead at midnight. When the Sun and Moon lie at right angles, the Moon is lighted from one side. We see a half-moon, and the Moon rises and sets six hours before (when waning) or after (when waxing) the Sun. When the Sun and Moon lie in the same direction, the far side of the Moon is lighted and our side is dark. It is new moon. The Sun and Moon will rise and set together, and the thin faint crescent of the new moon will be lost in the Sun's glare.

Tides are caused by the gravitational pull of the Sun and Moon on the waters of the oceans. The Moon has a much stronger effect than the Sun, and there will always be a high tide on the side of the Earth toward the Moon, and another on the side opposite the Moon, or two tides per day. When the Moon is new or full, the Sun and Moon combine to produce very high tides. During half-moons, the Sun and Moon pull at the right angles to each other and the tidal forces are weaker. Because it takes time to get an ocean full of water moving, the actual high tide will lag behind the Moon by some odd hours. The "solunar" charts are used by hunters and fishermen to predict peak periods of game and fish activity are simply charts of this tidal force.

If a full moon occurs exactly on one of the nodes, the Earth's shadow will fall across the Moon, and there will be a lunar eclipse. Because the Earth's shadow is larger than the Moon, a lunar eclipse is visible from anywhere on the night side of the Earth and may last several hours. If a new moon occurs exactly on one of the nodes, the Moon's shadow will fall across part of the Earth, and there will be a solar eclipse.

Since the Moon's shadow is much smaller than the Earth, the eclipse will only be seen in certain areas, and the alignment of Earth, Sun and Moon must be much more precise than for a lunar eclipse.

How could the ancient Britons have used Stonehenge as a computer to simulate this model of the cosmos? The secret lies in the circle of 56 Aubrey holes. Consider this circle to represent the fixed background of stars. Now place four large stones on the Aubrey holes for markers; one for the Sun, one for the Moon and two opposite each other for the nodes of the moon. Move the stones according to the following rules:

1. Move the moonstone one hole counter-clockwise every morning and evening.
2. Move the sunstone one hole counter-clockwise every 6½ days, or once for every 13 moves of the moonstone.
3. Move the nodestones one hole clockwise 3 times a year.

These simple rules approximate the motions of Sun, Moon and nodes but allow the stones to eventually drift out of position. The nodestones will stay in step for several centuries, and can be reset at leisure by observations or by waiting for an eclipse. The sunstone will drift out of position in a year or two, but can be reset at each summer and winter solstice. If one solstice is missed because of bad weather, the position will still be reasonably accurate. The moonstone will drift out of position most quickly — about four holes in three months. It can be reset at each new moon, but it is more likely an auxiliary rule was used, in the same way that we adjust our calendar during leap years. If the moonstone is moved one extra hole at each new moon, and moved one more extra hole at each solstice and equinox (that is, four times a year or every three months), the moonstone will stay in position for at least a year. Through a combination of ritually applied rules and careful astronomical sightings, it would have been possible to maintain an accurate model of solar and lunar movements, even when the skies were clouded for months at a time. However, if the realtime settings of the markers were lost, it might take several generations to restore them. Now we begin to see why Stonehenge was built of 30-ton stones, and why the grounds were considered sacred!

Let's run through a modern example to see how this computer might have been used, and how useful it might have been. The program in The Program Listing was used to determine the setting of the marker stones for March 13, 1979 (see Figure 3). The sun, moon and node stones are all aligned along holes 16 to 44. It is full moon and there will be a lunar eclipse (not visible from the United States). The Moon will rise just at sunset and reach its highest point at midnight. The tidal force will peak at noon and midnight, and those will be the best hours for hunting and fishing. High tides will arrive at Philadelphia about 1:30 a.m. and p.m. Because it is full moon, these will be relatively strong tides.

Figure 4 shows the settings for March 22, nine days later. The sunstone has just moved into hole 14, and it is the spring equinox. The moonstone is in hole 24, past half and waning. The Moon will rise after midnight, and the tidal forces will peak after dawn, with the high tides about 9:00 a.m. The tides will be increasing in force.

Figure 5 shows the situation on April 11th, twenty days later. The Moon has gone past new and back to full. This is the first full moon since the spring equinox. The calendar shows that the 11th is a Wednesday. The 15th of April is the first Sunday followed the first full moon, following the spring equinox, and it will be Easter Sunday!

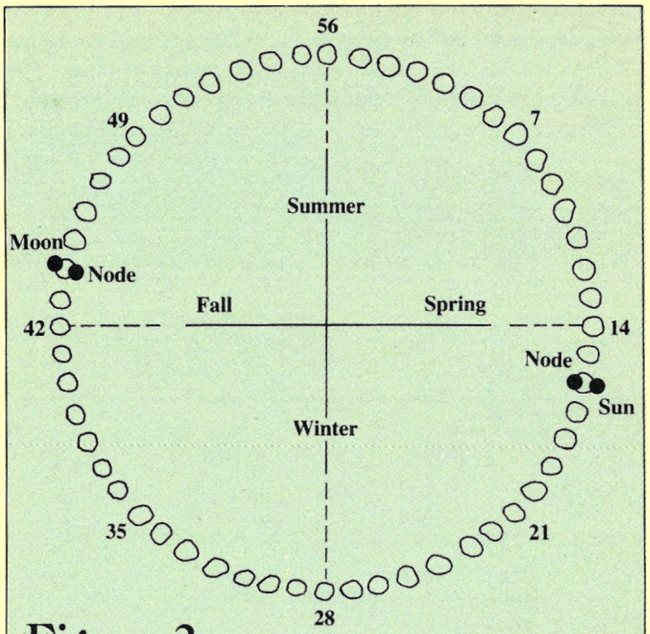


Figure 3 Marker settings for March 13, 1979

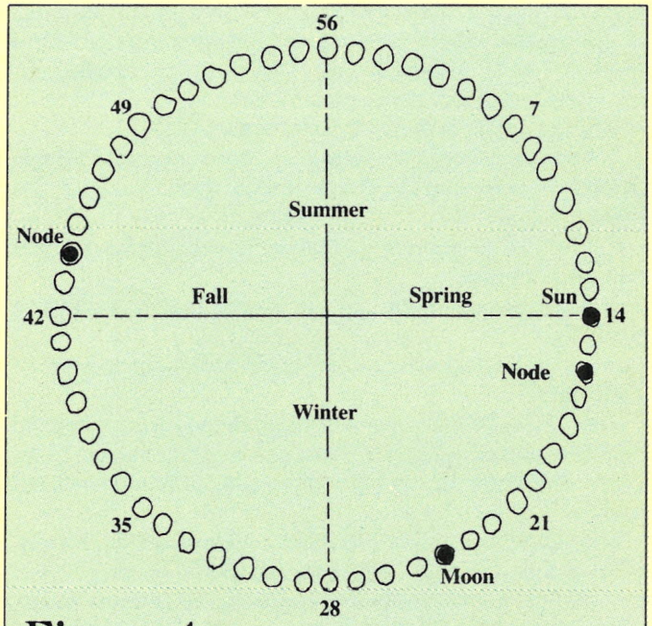


Figure 4 Marker settings for March 22, 1979

Returning to the (invisible) eclipse of March 14th, how can you know in advance whether a particular eclipse will be visible from a particular location? By taking careful sightings of the direction and time between sunset and moonrise, the uncertainty can be minimized, but not eliminated. However, there is a more nearly foolproof method. Some 6585.3 days, or exactly 223 full moons, after an exclipse there will be another eclipse of the same kind, about seven hours later in the day. If you knew of the eclipse occurring on August 26, 1961, you could be reasonably sure of seeing the one on September 6, 1979. Knowing that the eclipse of September 6th occurred near dawn, you would expect the eclipse of September 16, 1997 to occur during daylight and be invisible.

This cycle, called the saros, is pure coincidence. It could not be inferred or predicted and would not exist at all if the periods of Sun, Moon, or nodes were even slightly different. But if someone at Stonehenge were recording eclipses and keeping count of full moons, they would soon notice the cycle. At least 56 years of records are necessary to make

reasonably reliable predictions, and a great deal of effort must have gone into keeping tables of known eclipses. Lunar eclipse prediction should have been nearly perfect. The accurate prediction of solar eclipses was beyond their skill,

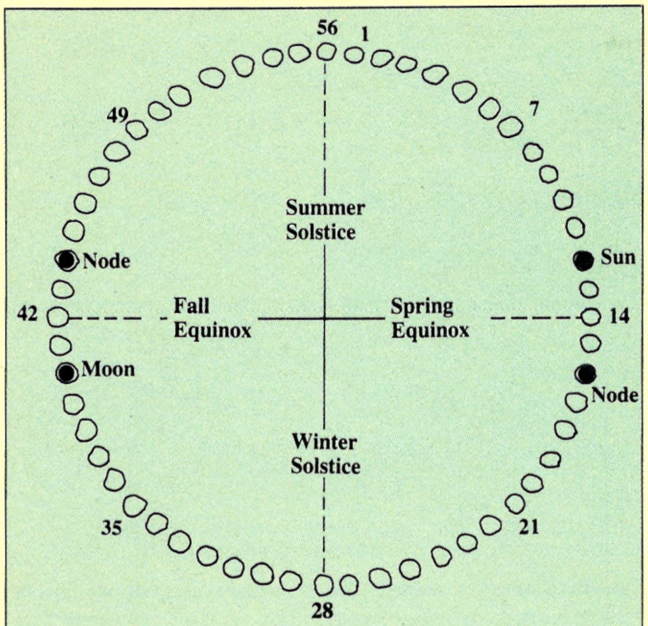


Figure 5 Marker settings for April 11, 1979

but they would have had warning of each one that actually occurred.

Why predict eclipses at all? The usual explanation if that eclipses were a terrifying sight to a primitive, superstitious people, and that they wanted advance warning of the catastrophe. Being able to predict eclipses would certainly have increased the prestige of priesthood. But we've seen that the builders of Stonehenge were no fools — it could be that they eagerly anticipated eclipses as an opportunity to precisely reset the markers in their megalithic computer and confirm the model built up so carefully from inference, observation and years of hard work!

The program in the Program Listing can be used to initialize a model of Stonehenge for experimentation and to compute the interval of 223 full moons for eclipse prediction. It does not correct for longitude or time of day and shouldn't be used as a substitute for an astronomical ephemeris, but the results should be within one day or one hole. The only other apparatus needed to experiment with a Stonehenge-type computer are a large cardboard chart of the 56 Aubrey holes, some coins for markers, and a calendar showing solstices, equinoxes and phases of the Moon. Eclipses are usually confirmed by the news media a few days ahead of time.

You can get the most out of a model by leaving it set up for a period of months and moving the markers each day, noticing the correlation between the model and the night sky. A classroom model maintained for an entire school year would make an excellent earth science project.

The routines in the Program Listing can also be used as the nucleus for a more ambitious program that graphically displays the position of each marker stone on a diagram of Stonehenge, possibly permitting you to step forward in time while animated markers move through their cycles.

In any case, when you get to thinking that your own computer is too slow or has too little memory for any important application, think about those Stone Age priests carrying boulders around a hundred meter circle to enter data into a computer built of 30-ton silicon "chips." Stonehenge may be classic example of clever programming overcoming hardware limitations!

Program Notes

This program was written in Programma version 2.0 BASIC for the Sphere System 3. Every effort was made to avoid special instructions and non-standard forms, so the program should translate into other BASICs with a minimum of debugging. The PRINT statements are formatted for a 32-character by 16-line screen. This program contains routines to perform five major functions:

1. Input a date in the form year, month, day.
2. Convert a date in the form year, month, day into a single number representing the count of days from December 31, 1949.
3. Convert a date in single-number form back into the form year, month, day.
4. Compute the dates 223 full moons before and after a given date.
5. Compute the settings of the markers at Stonehenge for a given date.

Lines 50 to 260 are the main program loop. Lines 60 to 80 clear the screen, and lines 90 to 190 print the menu. Lines 200 to 260 input the option number and jump to the desired section.

The subroutine at line 270 inputs a date as three numbers: Y, M and D. Line 310 makes the number Y an integer, and lines 320, 330 check to see that it is within the allowed range. Lines 340 to 360 will input the year again if Y is outside the range. The month and day are input in the same way.

The routine at line 570 computes a single index number, I, which is the number of days between December 31, 1949, and the date in Y, M and D. Line 590 multiplies the years by 365. Line 600 adds one day for every fourth year (leap years), and lines 610, 620 correct for the year 2000 (not a leap year). Lines 630 to 880 add the proper number of days for each month up to M. Lines 1050, 1060 check for the 29th day of February during leap years. Lines 890 to 920 add the days, D, and a small decimal to prevent the index number from computing to NNNN.9999999.

The subroutine at line 930 takes an index number I and returns a date in the form Y, M, D. Lines 940 to 970 put initial value into Y, M, D and transfer I to a temporary variable so that I is not lost during computation. Lines 980 to 1080 are a loop that subtract 365 from the index and add 1 to the year until the index is less than 365 (or 366 for a leap year). Lines 1090 to 1450 make a special case of each month, subtracting the correct number of days from the index, adding 1 to the month, and checking to see if there are enough days left for the next month. Lines 1460 to 1530 set the day equal to the number remaining in the index, and check for February 29th on leap years.

Lines 1550 to 1810 compute the saros in a straightforward way. The index for a known eclipse is computed (line 1610) and saved in I1 (line 1620). Then 6585 days are subtracted from the index and Y, M and D are computed for the previous

Sample Run

```
DO YOU WANT TO:

 0. EXIT TO BASIC?

 1. COMPUTE THE SETTING FOR A
     GIVEN DATE?

 2. COMPUTE THE SAROS FOR A
     KNOWN ECLIPSE?

ENTER OPTION (0-2) 1

ENTER THE DATE TO WHICH YOU WANT
 THE STONES SET--

YEAR? 1979
MONTH? 9
DAY? 6

SUNSTONE MOVED INTO HOLE  45
 ON DATE 1979 9 1

MOON STONE AT HOLE 17

N  STONE AT HOLE 17
N' STONE AT HOLE 45
N STONES MOVED 1979 9 1

HIT RETURN TO CONTINUE--

     SOFTWARE STONEHENGE

DO YOU WANT TO:

 0. EXIT TO BASIC?

 1. COMPUTE THE SETTING FOR A
     GIVEN DATE?

 2. COMPUTE THE SAROS FOR A
     KNOWN ECLIPSE?

ENTER OPTION (0-2) 2

ENTER THE DATE OF THE KNOWN
 ECLIPSE--

YEAR? 1979
MONTH? 9
DAY? 6

KNOWN ECLIPSE OCCURRED:
     1979 9 6

THE PREVIOUS ECLIPSE IN THIS
 SAROS OCCURRED:
     1961 8 26

THE NEXT ECLIPSE IN THIS SAROS
 WILL OCCUR:
     1997 9 16

HIT RETURN TO CONTINUE--
```

eclipse in that saros (lines 1660 to 1720). Lines 1730 to 1780 find the next eclipse in the cycle. The original index is recovered from I1 and 6585 days are added to it (line 1740). Y, M, D are found as before. Lines 1790, 1810 keep the dates from scrolling off the screen before they can be read.

Lines 1820 to 2360 compute the proper setting for the sun, moon and node stones for any given date between January 1, 1950, and December 31, 2099. Lines 1820 to 1910 get the date and compute its index number. Again, the index is saved in I1,this timeY,M and D are also saved in Y1, M1 and D1. Lines 1920 to 2040 find the sunstone setting. Y is set to 1950 (line 1930) because the sunstone setting is the same for a given date in any year. In line 1950, 192.1 represents the offset between January 1, the beginning of our calendar year, and Aubrey hole 1 at Stonehenge (marking midsummer's day). Dividing by 365 gives the index in years, and multiplying by 56 gives the proper hole number in I. In line 1960, I is subtracted from 56 because the Aubrey holes are numbered clockwise, and the sunstone is rotating counterclockwise. Line 1990 computes the date on which the sunstone was last moved; this eliminates a 6.5-day uncertainty in the sun's position. Lines 2050 to 2150 compute the position of the moonstone. Y, M and D are restored (lines 2060 to 2080) and the index is computed and stored (lines 2090, 2100). Line 2110 represents an offset for January 1, 1950. Line 2120 finds the days in the current lunar cycle (I reduced modulo 27.3217), line 2130 converts days into number of holes and reverses the direction of rotation. Lines 2160 to 2320 find the position of the nodestones. The index is recovered and an offset added to it (line 2170). The index is reduced modulo 6797.1 (lines 2180, 2190). The index is converted from days to number of holes away from N (lines 2230, 2250). Again, lines 2340, 2350 keep the information from scrolling away before it can be read. □

For Further Reading

Stonehenge

Hoyle, Fred, Sir. *On Stonehenge,* W.F. Freeman and Company, 1977

Hawkins, Gerald S. and White, John B. *Stonehenge Decoded,* Dell Publishing Company, Inc., 1965

Lunar Eclipses

Satterthwaite, Gilbert E. *Encyclopedia of Astronomy,* page 108, St. Martin's Press, N.Y., 1971

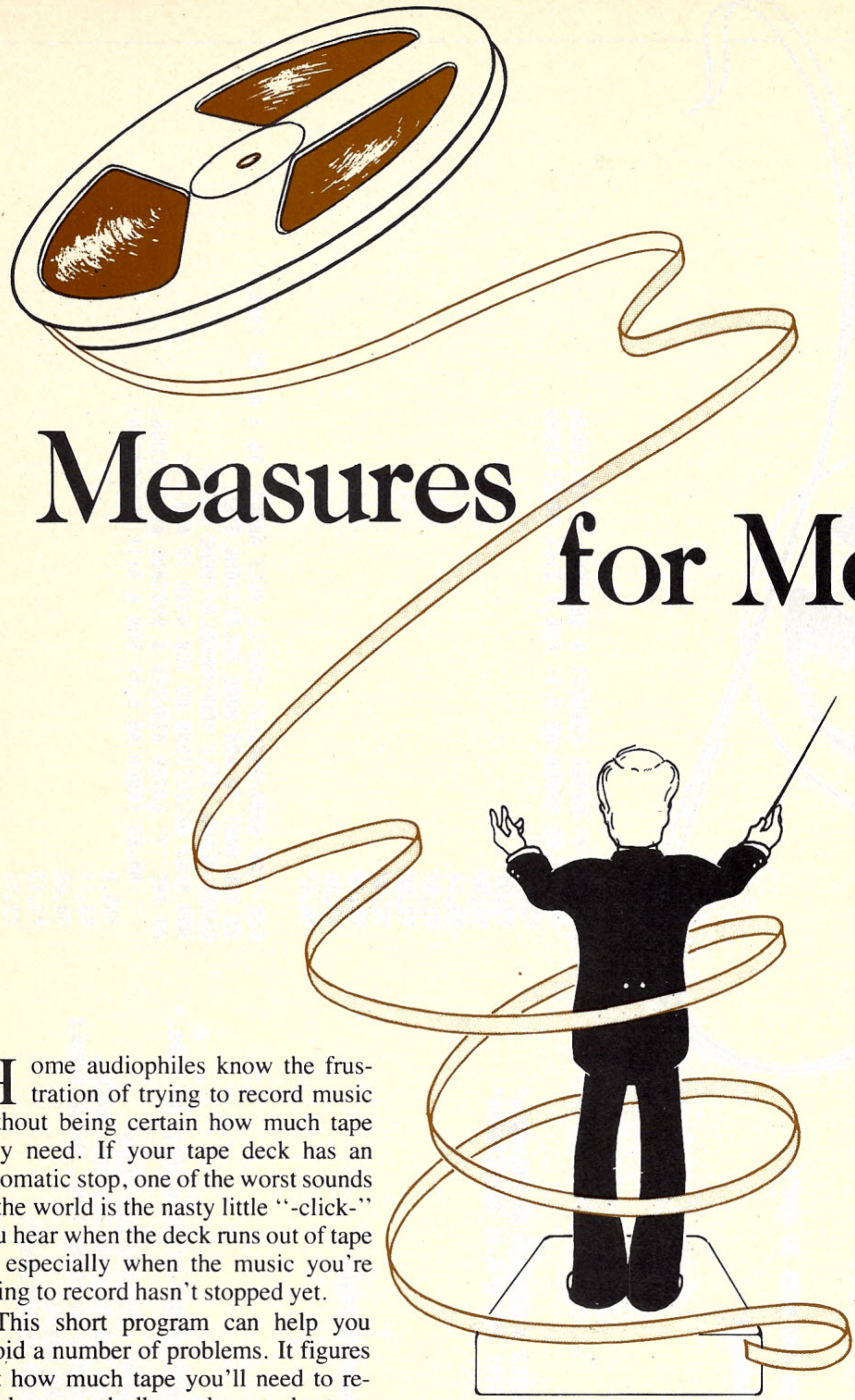
Solar Eclipses

The Encyclopedia Americana, volume 9: page 586, Danbury, Conn., 06816

Program Listing

```
10  REM -----"STONEHENGE SIMULATION PROGRAM"
20  REM -----"WRITTEN IN PROGRAMMA 2.0 BASIC FOR SPHERE SYS 3"
30  REM -----"1979 AUGUST 6 DJB"
40  REM "VERS. 10 ON H2 @ 37"
50  REM -----"MAIN LOOP BEGINS HERE"
60  FOR A=1 TO 16
70  PRINT
80  NEXT A
90  PRINT "      SOFTWARE STONEHENGE"
100  PRINT
110  PRINT "DO YOU WANT TO: "
120  PRINT
130  PRINT " 0. EXIT TO BASIC?"
140  PRINT
150  PRINT " 1. COMPUTE THE SETTING FOR A        GIVEN DATE?"
160  PRINT
170  PRINT " 2. COMPUTE THE SAROS FOR A          KNOWN ECLIPSE?"
180  PRINT
190  PRINT "ENTER OPTION (0-2)";
200  INPUT N
210  PRINT
220  PRINT
230  IF N=1 GOTO 1820
240  IF N=2 GOTO 1550
250  IF N<>0 GOTO 50
260  END
270  REM -----"SUBROUTINE TO INPUT DATE Y,M,D"
280  PRINT
290  PRINT "YEAR?";
300  INPUT Y
310  LET Y= INT( Y+0.001)
320  IF Y<1950 GOTO 340
330  IF Y<2100 GOTO 370
340  PRINT
```

```
350  PRINT "(1950 TO 2100 ONLY)"
360  GOTO 270
370  PRINT "MONTH?";
380  INPUT M
390  LET M= INT( M+0.001)
400  IF M<1 GOTO 420
410  IF M<13 GOTO 460
420  PRINT
430  PRINT "(MONTH FROM 1 TO 12)"
440  PRINT
450  GOTO 370
460  PRINT "DAY?";
470  INPUT D
480  LET D= INT( D+0.001)
490  IF D<1 GOTO 510
500  IF D<32 GOTO 550
510  PRINT
520  PRINT "(DAY FROM 1 TO 31)"
530  PRINT
540  GOTO 460
550  PRINT
560  RETURN
570  REM -----"SUBROUTINE TO CALCULATE INDEX I FROM DATE Y,M,D"
580  REM --"FIRST GET YEAR"
590  LET I=(Y-1950)*365
600  LET I=I+ INT( (Y-1949)/4+0.001)
610  IF Y<=2000 GOTO 630
620  LET I=I-1
630  REM --"NOW GET MONTH"
640  IF M=1 GOTO 890
650  LET I=I+31
660  IF M=2 GOTO 890
670  LET I=I+28
680  IF  INT( Y/4+0.001)*4<>Y GOTO 710
690  IF Y=2000 GOTO 710
700  LET I=I+1
710  IF M=3 GOTO 890
720  LET I=I+31
730  IF M=4 GOTO 890
740  LET I=I+30
750  IF M=5 GOTO 890
760  LET I=I+31
770  IF M=6 GOTO 890
780  LET I=I+30
790  IF M=7 GOTO 890
800  LET I=I+31
810  IF M=8 GOTO 890
820  LET I=I+31
830  IF M=9 GOTO 890
840  LET I=I+30
850  IF M=10 GOTO 890
860  LET I=I+31
870  IF M=11 GOTO 890
880  LET I=I+30
890  REM --"NOW GET DAY"
900  LET I=I+D
910  LET I=I+0.01
920  RETURN
930  REM -----"SUBROUTINE TO CALCULATE DATE Y,M,D FROM INDEX I"
940  LET A=I
950  LET Y=1950
960  LET M=1
970  LET D=1
980  REM --"LOOP HERE FOR YEAR"
990  IF  INT( (Y+1)/4+0.01)*4<>Y+1 GOTO 1020
1000  IF A<=366 GOTO 1090
1010  GOTO 1030
1020  IF A<=365 GOTO 1090
1030  LET A=A-365
1040  LET Y=Y+1
1050  IF  INT( Y/4+0.01)*4<>Y GOTO 1080
1060  IF Y=2000 GOTO 1080
1070  LET A=A-1
1080  GOTO 980
1090  REM --"DETERMINE MONTH"
1100  IF A<=31 GOTO 1460
1110  LET A=A-31
1120  LET M=M+1
1130  IF A<=29 GOTO 1460
1140  LET A=A-28
1150  LET M=M+1
1160  IF  INT( Y/4+0.01)<>Y GOTO 1190
1170  IF Y=2000 GOTO 1190
1180  LET A=A-1
1190  IF A<=31 GOTO 1460
1200  LET A=A-31
1210  LET M=M+1
1220  IF A<=30 GOTO 1460
1230  LET A=A-30
1240  LET M=M+1
1250  IF A<=31 GOTO 1460
1260  LET A=A-31
1270  LET M=M+1
1280  IF A<=30 GOTO 1460
1290  LET A=A-30
1300  LET M=M+1
1310  IF A<=31 GOTO 1460
1320  LET A=A-31
1330  LET M=M+1
1340  IF A<=31 GOTO 1460
1350  LET A=A-31
1360  LET M=M+1
1370  IF A<=30 GOTO 1460
1380  LET A=A-30
1390  LET M=M+1
1400  IF A<=31 GOTO 1460
1410  LET A=A-31
1420  LET M=M+1
1430  IF A<=30 GOTO 1460
1440  LET A=A-30
1450  LET M=M+1
1460  REM --"NOW THE DAY"
1470  LET D=A
1480  IF M<>2 GOTO 1540
1490  IF D<>29 GOTO 1540
1500  IF Y=2000 GOTO 1520
1510  IF  INT( Y/4+0.01)*4=Y GOTO 1540
1520  LET M=3
1530  LET D=1
1540  RETURN
1550  REM -----"THIS SECTION COMPUTES THE SAROS"
1560  PRINT
1570  PRINT
1580  PRINT "ENTER THE DATE OF THE KNOWN      ECLIPSE--"
1590  GOSUB 270
1600  PRINT
1610  GOSUB 570
1620  LET I1=I
1630  PRINT "KNOWN ECLIPSE OCCURRED:"
1640  PRINT "     ";Y;M;D
1650  PRINT
1660  REM --"FIND PREVIOUS ECLIPSE"
1670  LET I=I-6585
1680  IF I<1 GOTO 1730
1690  GOSUB 930
1700  PRINT "THE PREVIOUS ECLIPSE IN THIS     SAROS OCCURRED:"
1710  PRINT "     ";Y;M;D
1720  PRINT
1730  REM --"FIND NEXT ECLIPSE"
1740  LET I=I1+6585
1750  GOSUB 930
1760  PRINT "THE NEXT ECLIPSE IN THIS SAROS   WILL OCCUR:"
1770  PRINT "     ";Y;M;D
1780  PRINT
1790  PRINT "HIT RETURN TO CONTINUE--";
1800  INPUT A
1810  GOTO 50
1820  REM -----"THIS SECTION SETS STONES FOR DATE Y,M,D"
1830  PRINT
1840  PRINT
1850  PRINT "ENTER THE DATE TO WHICH YOU WANT THE STONES SET--"
1860  GOSUB 270
1870  LET Y1=Y
1880  LET M1=M
1890  LET D1=D
1900  GOSUB 570
1910  LET I1=I
1920  REM --"SUN STONE FIRST"
1930  LET Y=1950
1940  GOSUB 570
1950  LET T=(I+192.1)/365*56
1960  LET S=56- INT( T)
1970  IF S>=1 GOTO 1990
1980  LET S=S+56
1990  LET I=I1-((T- INT( T))*6.5)
2000  GOSUB 930
2010  PRINT
2020  PRINT "SUNSTONE MOVED INTO HOLE ";S
2030  PRINT " ON DATE";Y;M;D
2040  PRINT
2050  REM --"SET MOON STONE"
2060  LET Y=Y1
2070  LET M=M1
2080  LET D=D1
2090  GOSUB 570
2100  LET I1=I
2110  LET I=I-2.3
2120  LET M=I- INT( I/27.3217)*27.3217
2130  LET M=56- INT( M*56/27.32)
2140  PRINT "MOON STONE AT HOLE";M
2150  PRINT
2160  REM -----"NOW SET N AND N'"
2170  LET I=I1+4937
2180  LET T= INT( I/6797.1)*6797.1
2190  LET N=I-T
2200  LET N= INT( N*56/6798)
2210  IF N>=1 GOTO 2230
2220  LET N=N+56
2230  LET N1=N+28
2240  IF N1<=56 GOTO 2270
2250  LET N1=N1-56
2260  PRINT
2270  PRINT "N  STONE AT HOLE";N
2280  PRINT "N' STONE AT HOLE";N1
2290  LET Y=Y1
2300  LET M= INT( (M1-1)/4)*4+1
2310  LET D=1
2320  PRINT "N STONES MOVED";Y;M;D
2330  PRINT
2340  PRINT "HIT RETURN TO CONTINUE--";
2350  INPUT A
2360  GOTO 50
```



Measures for Measures

BY ROD MORGAN

Home audiophiles know the frustration of trying to record music without being certain how much tape they need. If your tape deck has an automatic stop, one of the worst sounds in the world is the nasty little "-click-" you hear when the deck runs out of tape — especially when the music you're trying to record hasn't stopped yet.

This short program can help you avoid a number of problems. It figures out how much tape you'll need to record a record album, how to best rearrange selections to fit the tape you're using, and how much extra time you'll have left at the end of the tape. It also lets you avoid the hassle of "Base 60" math, since it converts seconds into minutes and seconds quickly and easily.

The Sample Runs illustrate some of these applications. The soundtrack album for the TV show "Battlestar Galactica" contains 16 selections, which immediately makes you think it's longer than the average album. But timing side one shows there's less than 20 minutes of music on the side. Adding the time of side one to the times of the selections on side two, we get the total, and find that the entire album will fit on one side of a 90-minute cassette — with enough time left over for a disco version of the theme song by another artist. (The two sides will also fit comfortably on a C-45 cassette, if you prefer.)

The next few runs involve the last three albums recorded by Henry Mancini: "The Cop Show Themes", "Mancini's Angels" and "The Theme Scene". Assume we want to fit at least two of these onto a C-90 cassette. We time "The Cop Show Themes", which is only 24 minutes and 50 seconds long. We then add the times for "Mancini's Angels", still coming up with a total time of less than one hour. Adding the selections on the "Theme Scene" album, we discover all three will fit onto the same 90-minute cassette.

A few more runs will show us the best way to fit the albums onto the cassette. Timing "The Cop Show Themes" and side one of "Mancini's Angels", we have a total of about 41 minutes of music. Adding that total to the time of the first selection on side two of "Mancini's Angels", we have the best length for side one of the cassette. Then double-checking the remaining selections' times, we find that the other tracks will fill out side two. This arrangement will also fit the three albums in chronological order of release, which some tape buffs find desirable.

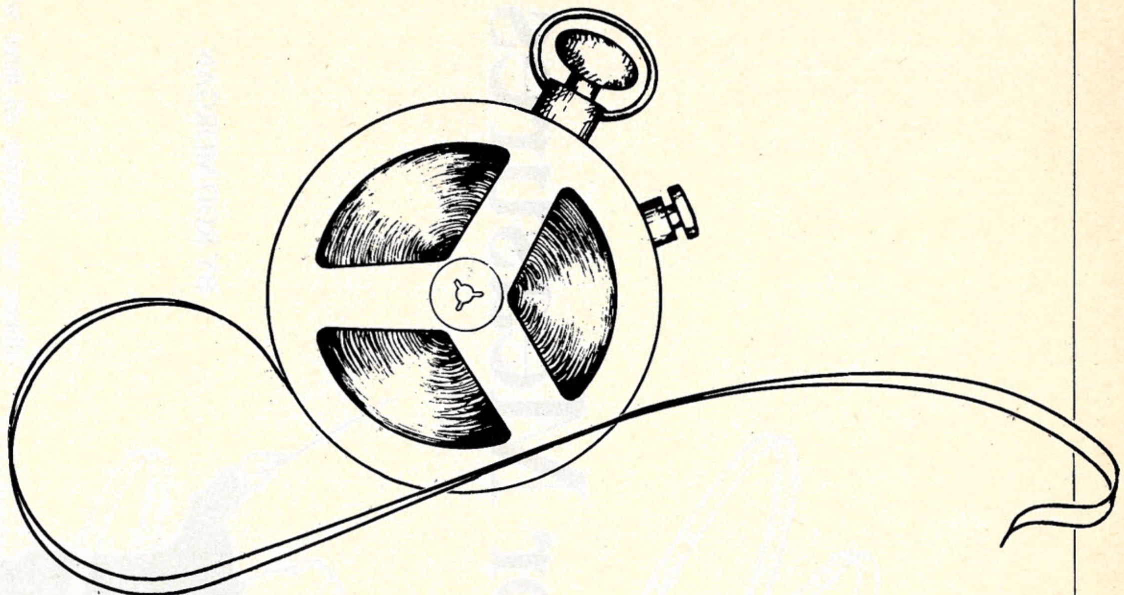
This program can also help you arrange "sampler" cassettes or put together your own "radio shows".

Incidentally, current interpretations of the copyright laws judge that this type of taping, for personal use, of recordings you already own is perfectly legal. Copying a friend's LP for yourself is technically illegal; and, of course, making tape copies of an album for re-sale can subject you to federal prosecution. So, while using this program, keep your nose clean! □

Illustration by David Bastille

Program Listing

```
00100  REM     RECORDING TIMER
00110  REM     PROGRAM BY ROD MORGAN
00120  REM                P O BOX 44
00130  REM                HERNDON, VA 22070
00140  REM
00150  REM     THIS PROGRAM ACCEPTS A NUMBER OF TIMINGS
00160  REM     AND TOTALS THEM.  THIS SPECIFIC APPLICATION
00170  REM     IS FOR TAPING RECORD ALBUMS, ETC.
00180  REM     MORE THAN 25 LISTINGS CAN BE ADDED BY
00190  REM     CHANGING LINE 270.
00200  REM
00210  REM     ALBUM TITLE MEMO, NUMBER OF SELECTIONS
00220     PRINT "ENTER ALBUM TITLE";
00230       INPUT R$
00240     PRINT "HOW MANY SELECTIONS DO YOU WISH TO ADD";
00250       INPUT T
00260  REM     DIMENSION AND CLEAR VARIABLES
00270     DIMENSION M(25), S(25)
00280     M1=0
00290     S1=0
00300        FOR C=1 TO T
00310        M(C)=0
00320        S(C)=0
00330        NEXT C
00340  REM     ENTER TIMINGS
00350     PRINT "AT THE ´?´ ENTER MINUTES & SECONDS AS FOLLOWS:"
00360     PRINT "      MM,SS (SUCH AS 5,11 FOR A 5:11 SELECTION)"
00370        FOR C=1 TO T
00380        INPUT M(C), S(C)
00390        M1=M1+M(C)
00400        S1=S1+S(C)
00410        NEXT C
00420  REM     CONVERT SECONDS TO MINUTES/SECONDS
00430     S1=S1/60
00440     M2=INT(S1)
00450     S1=S1-M2
00460     S1=S1*60
00470     M1=M1+M2
00480  REM     PRINT TITLE AND TOTALS
00490     PRINT "FOR ´";R$;"´ YOU´LL NEED ABOUT ";M1;":";S1;" OF TAPE."
00500  REM
00510  REM     THE PROGRAM ADVISES ´ABOUT´ SINCE ALBUM TIMES
00520  REM     GENERALLY DO NOT INCLUDE THE SILENCE BETWEEN TRACKS.
00530  REM     THIS WILL ADD A FEW SECONDS TO TOTAL TIME, WHICH SHOULD
00540  REM     BE CONSIDERED BY PROGRAM USERS.
00550     PRINT "DO YOU WANT TO ADD MORE SELECTIONS";
00560       INPUT J$
00570         IF LEFT$(J$,1)="Y" GOTO 100
00580     PRINT "OKAY - SEE YOU LATER."
00590  END
```



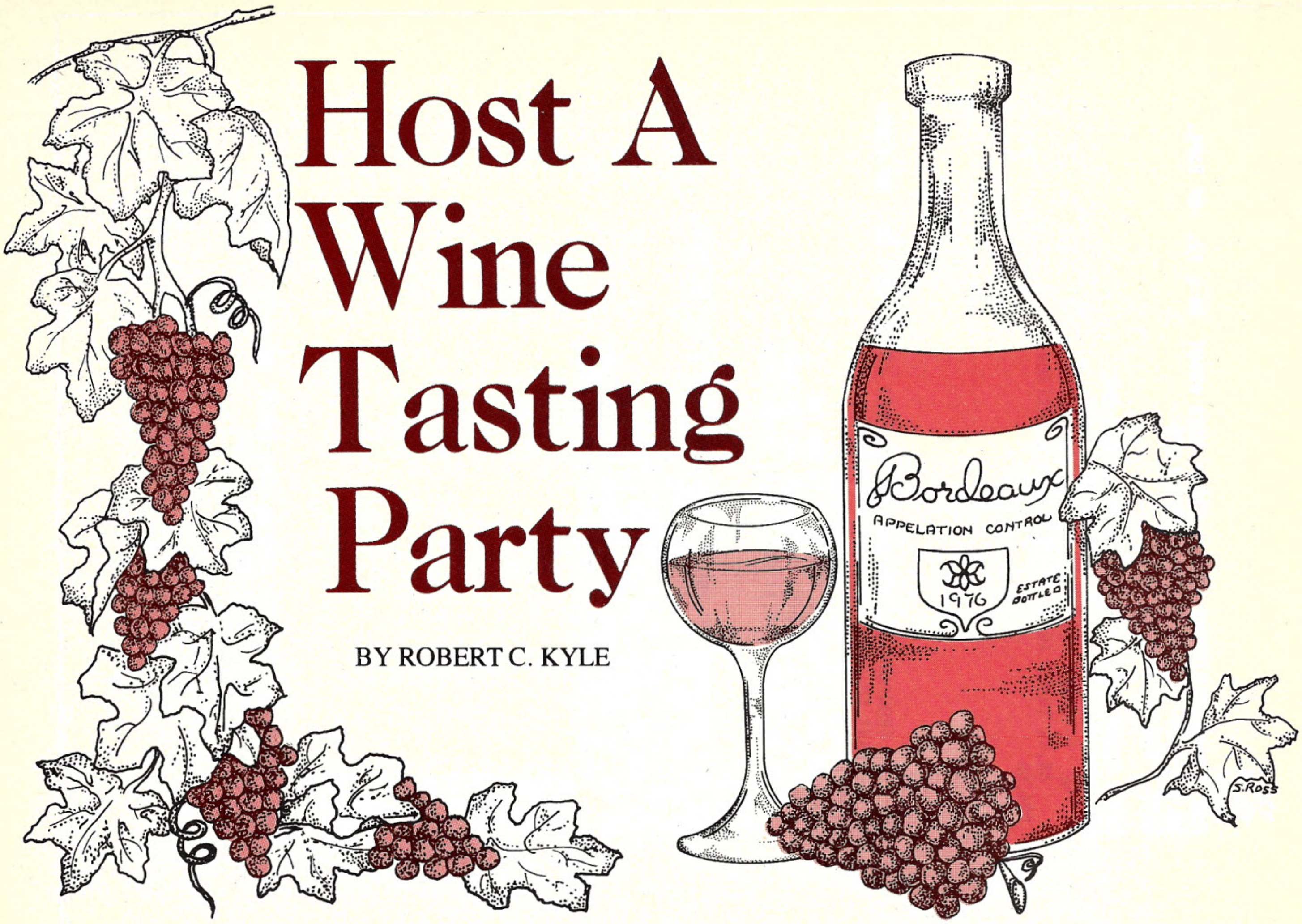
```
AT THE ´?´ ENTER MINUTES & SECONDS AS FOLLOWS:
     MM,SS (SUCH AS 5,11 FOR A 5:11 SELECTION)
 ?24,50
 ?3,01
 ?3,50
 ?3,14
 ?2,57
 ?2,58
 ?2,07
 ?2,48
 ?2,38
 ?6,37

FOR ´COP SHOW/ANGELS´ YOU´LL NEED ABOUT  55 : 0  OF TAPE.
DO YOU WANT TO ADD MORE SELECTIONS ?YES
ENTER ALBUM TITLE ?COP/ANGELS/THEME SCENE
HOW MANY SELECTIONS DO YOU WISH TO ADD ?11
AT THE ´?´ ENTER MINUTES & SECONDS AS FOLLOWS:
     MM,SS (SUCH AS 5,11 FOR A 5:11 SELECTION)
 ?55,0
 ?3,11
 ?4,03
 ?3,13
 ?3,31
 ?1,51
 ?2,52
 ?3,35
```

Sample Run

```
ENTER ALBUM TITLE ?GALACTICA SIDE ONE
HOW MANY SELECTIONS DO YOU WISH TO ADD ?8
AT THE '?' ENTER MINUTES & SECONDS AS FOLLOWS:
     MM,SS (SUCH AS 5,11 FOR A 5:11 SELECTION)
 ?1,28
 ?4,01
 ?2,49
 ?2,54
 ?3,24
 ?1,14
 ?1,42
 ?1,42
FOR 'GALACTICA SIDE ONE' YOU'LL NEED ABOUT  19 : 14.  OF TAPE.
DO YOU WANT TO ADD MORE SELECTIONS ?YES
ENTER ALBUM TITLE ?BATTLESTAR GALACTICA
HOW MANY SELECTIONS DO YOU WISH TO ADD ?9
AT THE '?' ENTER MINUTES & SECONDS AS FOLLOWS:
     MM,SS (SUCH AS 5,11 FOR A 5:11 SELECTION)
 ?19,14
 ?3,51
 ?2,57
 ?2,23
 ?1,52
 ?4,10
 ?2,41
 ?1,13
 ?1,06
FOR 'BATTLESTAR GALACTICA' YOU'LL NEED ABOUT  39 : 27.  OF TAPE.
DO YOU WANT TO ADD MORE SELECTIONS ?YES
ENTER ALBUM TITLE ?COP SHOW THEMES
HOW MANY SELECTIONS DO YOU WISH TO ADD ?8
AT THE '?' ENTER MINUTES & SECONDS AS FOLLOWS:
     MM,SS (SUCH AS 5,11 FOR A 5:11 SELECTION)
 ?1,58
 ?2,48
 ?1,55
 ?4,44
 ?5,03
 ?2,51
 ?2,17
 ?3,14
FOR 'COP SHOW THEMES' YOU'LL NEED ABOUT  24 : 50.  OF TAPE.
DO YOU WANT TO ADD MORE SELECTIONS ?YES
ENTER ALBUM TITLE ?COP SHOW/ANGELS
HOW MANY SELECTIONS DO YOU WISH TO ADD ?10
 ?3,10
 ?2,59
 ?2,47
FOR 'COP/ANGELS/THEME SCENE' YOU'LL NEED ABOUT  86 : 12.  OF TAPE.
DO YOU WANT TO ADD MORE SELECTIONS ?NO
OKAY - SEE YOU LATER.

ENTER ALBUM TITLE ?COP SHOW/ANGELS SIDE ONE
HOW MANY SELECTIONS DO YOU WISH TO ADD ?6
AT THE '?' ENTER MINUTES & SECONDS AS FOLLOWS:
     MM,SS (SUCH AS 5,11 FOR A 5:11 SELECTION)
 ?24,50
 ?3,01
 ?3,50
 ?3,14
 ?2,57
 ?2,58
FOR 'COP SHOW/ANGELS SIDE ONE' YOU'LL NEED ABOUT  40 : 50.  OF TAPE.
DO YOU WANT TO ADD MORE SELECTIONS ?YES
ENTER ALBUM TITLE ?COP SHOW/ANGELS
HOW MANY SELECTIONS DO YOU WISH TO ADD ?2
AT THE '?' ENTER MINUTES & SECONDS AS FOLLOWS:
     MM,SS (SUCH AS 5,11 FOR A 5:11 SELECTION)
 ?40,50
 ?2,07
FOR 'COP SHOW/ANGELS' YOU'LL NEED ABOUT  42 : 57  OF TAPE.
DO YOU WANT TO ADD MORE SELECTIONS ?YES
ENTER ALBUM TITLE ?ANGELS/THEME SCENE
HOW MANY SELECTIONS DO YOU WISH TO ADD ?13
AT THE '?' ENTER MINUTES & SECONDS AS FOLLOWS:
     MM,SS (SUCH AS 5,11 FOR A 5:11 SELECTION)
 ?2,48
 ?2,38
 ?6,37
 ?3,11
 ?4,03
 ?3,13
 ?3,31
 ?1,51
 ?2,52
 ?3,35
 ?3,10
 ?2,59
 ?2,47
FOR 'ANGELS/THEME SCENE' YOU'LL NEED ABOUT  43 : 15  OF TAPE.
DO YOU WANT TO ADD MORE SELECTIONS ?NO
OKAY - SEE YOU LATER.
```

Host A Wine Tasting Party

BY ROBERT C. KYLE



Some computers not only play the role of financial managers and auxiliary homemakers, they can also step in as gracious hosts, especially at wine tasting parties.

First, visit your favorite beverage store; a good wine merchant will be more than happy to help you select the wines for your party. From my own experience I've found that tasting more than six wines becomes confusing to your guests; and more than ten guests makes the tasting scene a little too crowded. However, line 30 of the program allows for twelve people (A$) and eight wines (W$).

Wine tasting parties are great ice breakers and good intellectual mixers. Most people are hesitant about passing judgment on wines (until they see how easy it is) and are also afraid of computers. Therefore, the computer's statements at the beginning and at the end help make your guests a little less anxious. Having them enter their names into the computer as soon as they arrive is the best way I've found to make them feel more at ease with the machine.

Once the tasting progresses and the guests become more relaxed (due more to the wine), they become more proficient in entering data. The computer corrects them in a very nice and gentle manner to promote better relations between people and the mind machine.

I have to admit to my ego trip from seeing my non-computer friends marvel at the way the machine handles the entry of "all that data". After all, ten people scoring six wines is a lot of math!

The program, written for a TRS-80 Level II with 16K, will probably run on just about any machine. CHR$(23), used throughout the program, puts the video printout in the 32-characters-per-line mode. For the actual scoring I changed back to 64 characters per line. If you use the 32-character mode, you'll have to play around with the spacing so that the words won't spill over onto the next line.

The subroutine at line 300 calculates a running average of the scores of each wine and then prints them out at the end of the tasting. You could add a bubble sort subroutine at this point to print out the wine that scored first and then second and so on — even making the computer offer some statements about each wine of a certain rank.

Also note that the TRS-80 allows hitting the Enter key (carriage return) as a legal entry. If your machine does not allow this, use a variable entry instead.

With enough memory, you could have the computer store each person's score along with calculating the average for each wine. Then, in the final printout, your guests could see what influence their score had on the rating of the wines. This revelation is not really earth shattering, but will help them feel more comfortable when they order wines or find themselves confronted with a computer.

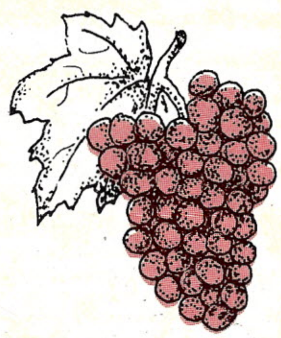
The main thing to watch out for is not to let anyone approach your machine with a wine glass in hand — even an empty one! One drop of the *vino* can be disastrous when you power up again. So keep your computer on a small table away from the wines and glasses.

Your guests should score each wine as they taste it. The first entry always takes the longest but you will find that as the night progresses, a person can enter the data in less than a minute. But, you should always stand by just in case. So it's best that you and the computer consume as little wine as possible; let your guests have all the fun. □

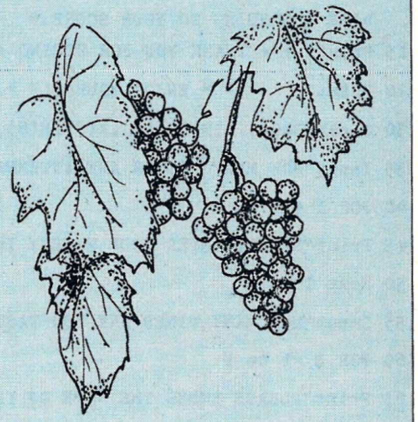
Illustrations by Susan Ross

Program Listing

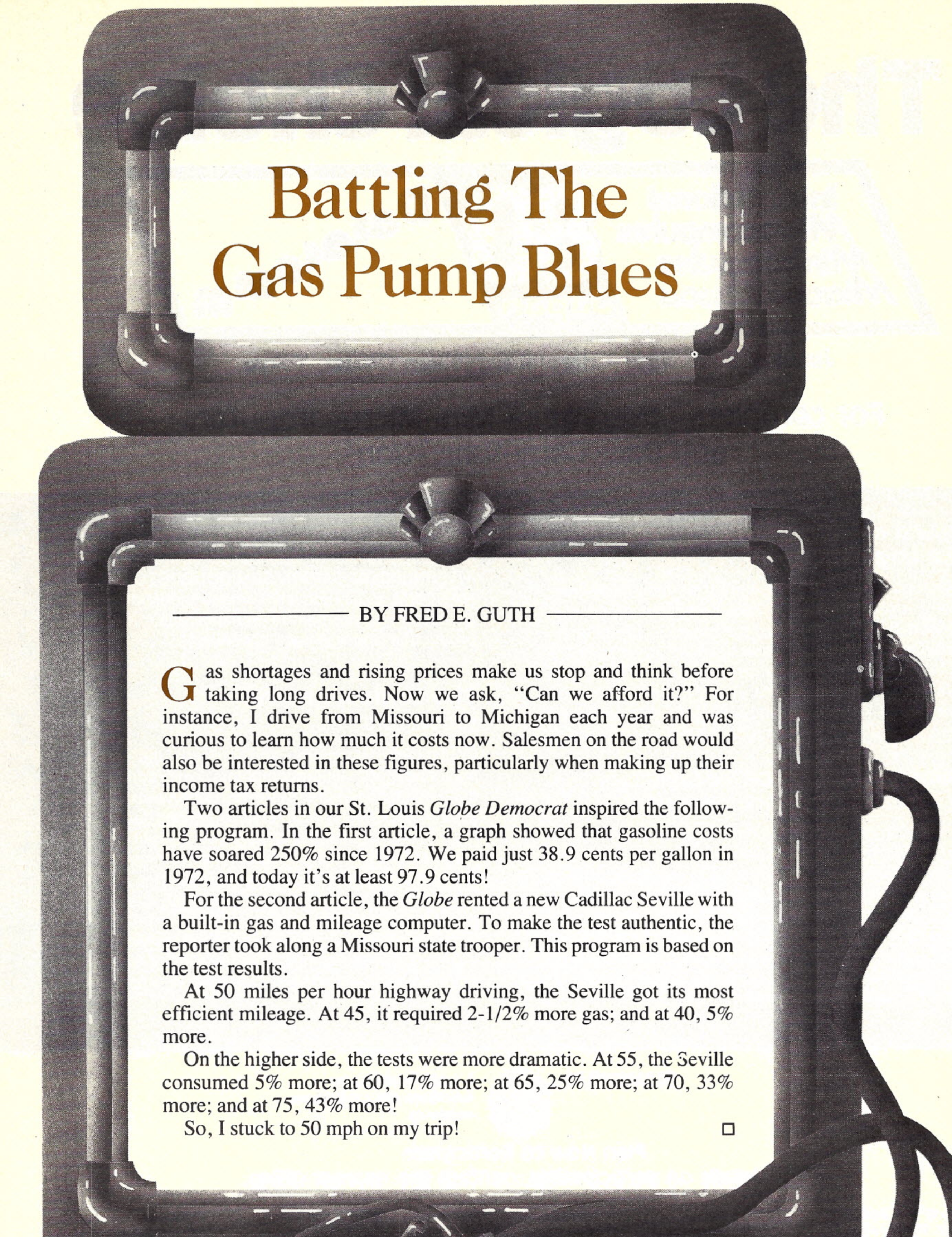
```
5 CLS: Print TAB(23)CHR$(23)"WINE TASTE"
10 Print"WELCOME TO MY WINE TASTING PARTY. YOU ARE INVITED
   TO TASTE EACH WINE AND THEN SCORE IT ACCORDING TO ITS
   AROMA,BOQUET,FLAVOR,BODY AND FINISH. I WILL RATE EACH
   WINE ACCORDING TO YOUR SCORE."
15 Print" + + THANK YOU FOR COMING + + "
20 Print"      * * ENJOY YOURSELF * * "
30 Clear 2000: DIM A$(12),A(8),H(8),W$(8)
35 Input"HOW MANY PEOPLE ARE ATTENDING";N
40 FOR I = 1 to N
45 Print"PLEASE ENTER YOUR NAME"; Input A$(I)
50 Next I
55 Input"HOW MANY WINES ARE YOU TASTING";P
60 FOR B =1 to P
65 Print"PLEASE ENTER THE NAME OF THE WINE":Input W$(B)
70 Next B
75 CLS: Print STRING$(10,"*");"WINE SCORE SHEET";
   STRING$(10,"*")
80 Input"ENTER THE NAME OF THE WINE YOU HAVE TASTED";T$
85 Input"WHO IS SCORING THE WINE";B$
90 FOR I =1 to N
95 IF B$=A$(I) THEN GOTO 115
100 Next I
105 Print"ENTER YOUR NAME AS YOU DID EARLIER THIS EVENING.
    TO SEE GUEST LIST - HIT *ENTER*":Input C:GOSUB 500
110 GOTO 85
115 GOSUB 200
120 Print B$;",YOU SCORED";S1;"FOR ";T$
125 Print"I WISH TO ADD YOUR DATA TO MY FILES. PLEASE HIT
    *ENTER*": Input C:GOSUB 300
130 Print"TYPE 1 TO CONTINUE SCORING WINES:TYPE 2 TO SEE
    FINAL GRAND AVERAGE OF WINES.":Input Y
135 IF Y=1 THEN 75
140 FOR B =1 To P
145 Print CHR$(23);W$(B),A(B)
150 Next B
155 Print"HAVE ALL WINES BEEN TASTED AND SCORES ENTERED?
    (1=YES:Ø=NO)":Input Y
160 IF Y=Ø THEN 75
165 CLS:Print CHR$(23):Print"THANK YOU FOR COMING"
170 Print"      * * GOOD DRINKING * * "
175 GOTO 999
200 FOR B=1 to P
205 IF T$=W$(B) THEN 230
210 Next B
215 Print"WINE IS NOT FILED AS ENTERED. HIT *ENTER* TO SEE
    CORRECT ENTRY":Input C:GOSUB 450
220 GOTO 80
230 Input"WHAT IS SCORE FOR AROMA (4 POINTS MAX!)";A1
231 Input"WHAT IS SCORE FOR BOQUET (4 POINTS MAX!)";B1
232 INPUT"WHAT IS SCORE FOR FLAVOR (7POINTS MAX!)";C1
233 Input"WHAT IS SCORE FOR BODY (3 POINTS MAX!)";D1
234 Input"WHAT IS SCORE FOR FINISH (2 POINTS MAX!)";E1
235 IF A1>4 OR b1>4 OR C1>7 OR D1>3 OR E1>2 THEN 236
236 Print"INPUT ERROR!! REDO!!":GOTO 230
240 S1=A1+B1+C1+D1+E1: M=10/S1
245 IF M>=.8 Print"VERY POOR WINE...SORRY ABOUT THAT!"
246 IF M<.8 and M>=.7 Print"NOT BAD! THERE ARE WINES
    THAT ARE WORSE."
247 IF M<.7 and M>=.6 Print"NICE LITTLE WINE. GOOD
    FOR EVERYDAY DRINKING."
248 IF M<.6 Print"* * EXCELLENT SELECTION * * SHOULD
    BE IN EVERY WINE CELLAR!!"
250 RETURN
300 FOR B = 1 to P
305 IF T$=W$(B) Then 315
310 Next B
315 ON B GOTO 320,340,350,360,370,380,390,400
320 K=K+1:L=L+S1
325 A(B)=L/K
330 Print"GRAND AVERAGE FOR ";T$;" IS";A(B)
335 Input"WHEN READY -HIT *ENTER*";C:RETURN
340 K1=K1+1:L1=L1+S1
345 A(B)=L1/K1:GOTO 330
350 K2=K2+1:L2=L2+S1
355 A(B)=L2/K2:GOTO 330
360 K3=K3+1:L3=L3+S1
365 A(B)=L3/K3:GOTO 330
370 K4=K4+1:L4=L4+S1
375 A(B)=L4/K4:GOTO 330
380 K5=K5+1:L5=L5+S1
385 A(B)=L5/K5:GOTO 330
390 K6=K6+1:L6=L6+S1
395 A(B)=L6/K6:GOTO 330
400 K7=K7+1:L7=L7+S1
405 A(B)=L7/K7:GOTO 330
450 FOR B= 1 to P
455 Print W$(B)
460 Next B
465 Input"WHEN READY -HIT *ENTER*";C:RETURN
500 FOR I = 1 to N
505 Print A$(I)
510 Next I
515 Input"WHEN READY -HIT *ENTER";C:RETURN
999 END
```



Sample Run



```
WELCOME TO MY WINE TASTING PARTY. YOU ARE INVITED TO TASTE EACH
WINE AND THEN SCORE IT ACCORDING TO ITS AROMA,FLAVOR,BOQUET,BODY
AND FINISH. I WILL RATE EACH WINE ACCORDING TO YOUR SCORE.
          + + THANK YOU FOR COMING + +
          * * ENJOY YOURSELF * *
HOW MANY PEOPLE ARE ATTENDING ? 2
PLEASE ENTER YOUR NAME
?ROBERT KYLE
PLEASE ENTER YOUR NAME
?JAMES BRENFIELD
HOW MANY WINES ARE YOU TASTING ? 2
PLEASE ENTER THE NAME OF THE WINE
?PAUL MASSON BURGUNDY
PLEASE ENTER THE NAME OF THE WINE
?NAVELLE BURGUNDY
             ********** WINE SCORE SHEET **********
ENTER THE NAME OF THE WINE YOU HAVE TASTED ? NAVELLE BURGUNDY
WHO IS SCORING THE WINE? BOB KYLE
ENTER YOUR NAME AS YOU DID EARLIER THIS EVENING. TO SEE GUEST
LIST -HIT *ENTER*?
    ROBERT KYLE

    JAMES BRENFIELD


WHO IS SCORING THE WINE? ROBERT KYLE

WHAT IS SCORE FOR AROMA (4 POINTS MAX!)? 2
WHAT IS SCORE FOR BOQUET (4 POINTS MAX!)? 3
WHAT IS SCORE FOR FLAVOR (7 POINTS MAX!)? 5
WHAT IS SCORE FOR BODY (3 POINTS MAX!)? 2
WHAT IS SCORE FOR FINISH (2 POINTS MAX!)? 2

NOT BAD! THERE ARE WINES THAT ARE WORSE

ROBERT KYLE YOU SCORED 14 FOR NAVELLE BURGUNDY

I WISH TO ADD YOUR DATA TO MY FILES. PLEASE HIT *ENTER* ?

GRAND AVERAGE FOR NAVELLE BURGUNDY IS 14

TYPE 1 TO CONTINUE SCORING WINES: TYPE 2 TO SEE FINAL GRAND
AVERAGE ? 1

ENTER THE NAME OF THE WINE YOU HAVE TASTED? PAUL MASSON BURGUNDY
WHO IS TASTING THE WINE? JAMES BRENFIELD

WHAT IS SCORE FOR AROMA (4POINTS MAX!)? 3
WHAT IS SCORE FOR BOQUET (4 POINTS MAX!)? 4
WHAT IS SCORE FOR FLAVOR (7 POINTS MAX!)? 6
WHAT IS SCORE FOR BODY (3 POINTS MAX!)? 3
WHAT IS SCORE FOR FINISH (2 POINTS MAX!)? 2

* * EXCELLENT SELECTION * * SHOULD BE IN EVERY WINE CELLAR!!
JAMES BRENFIELD YOU HAVE SCORED 18 FOR PAUL MASSON BURGUNDY

I WISH TO ADD YOUR DATA TO MY FILE. PLEASE HIT *ENTER*?

GRAND AVERAGE FOR PAUL MASSON BURGUNDY IS 18

TYPE 1 TO CONTINUE SCORING WINES: TYPE 2 TO SEE FINAL GRAND
AVERAGE? 2

     PAUL MASSON BURGUNDY       18
     NAVELLE BURGUNDY           14

HAVE ALL THE WINES BEEN TASTED AND SCORED ENTERED?(1=YES:Ø=NO)
? 1
               THANK YOU FOR COMING
               * * GOOD DRINKING * *

READY-
```



Battling The Gas Pump Blues

BY FRED E. GUTH

Gas shortages and rising prices make us stop and think before taking long drives. Now we ask, "Can we afford it?" For instance, I drive from Missouri to Michigan each year and was curious to learn how much it costs now. Salesmen on the road would also be interested in these figures, particularly when making up their income tax returns.

Two articles in our St. Louis *Globe Democrat* inspired the following program. In the first article, a graph showed that gasoline costs have soared 250% since 1972. We paid just 38.9 cents per gallon in 1972, and today it's at least 97.9 cents!

For the second article, the *Globe* rented a new Cadillac Seville with a built-in gas and mileage computer. To make the test authentic, the reporter took along a Missouri state trooper. This program is based on the test results.

At 50 miles per hour highway driving, the Seville got its most efficient mileage. At 45, it required 2-1/2% more gas; and at 40, 5% more.

On the higher side, the tests were more dramatic. At 55, the Seville consumed 5% more; at 60, 17% more; at 65, 25% more; at 70, 33% more; and at 75, 43% more!

So, I stuck to 50 mph on my trip! □

Illustration by Josh Randall

Sample Run 1972

THE TOTAL MILEAGE WILL BE - 825 MILES
AT 50 MPH THE MILES-PER-GALLONS ARE - 17 MPG
OIL CONSUMPTION IS - 2 QTS/M
AN AVERAGE GAL. OF GAS COSTS - $ .389
AVERAGE QT. OF OIL COSTS - $ .45

AVG. SPEED	EST. GAS USAGE GALS.	COST	EST. OIL USE QTS.	COST	TOTAL GAS/OIL	GAS STOPS	HOURS REQ'D.
40	50.96	$ 19.82	1.65	$ .74	$ 20.56	3	20.63
45	49.74	$ 19.35	1.65	$ .74	$ 20.09	3	18.33
50	48.53	$ 18.88	1.65	$ .74	$ 19.62	3	16.5
55	50.96	$ 19.82	1.65	$ .74	$ 20.56	3	15
60	56.78	$ 22.09	1.65	$ .74	$ 22.83	3	13.75
65	60.66	$ 23.6	1.65	$ .74	$ 24.34	4	12.69
70	64.54	$ 25.11	1.65	$ .74	$ 25.85	4	11.79
75	69.4	$ 27	1.65	$ .74	$ 27.74	4	11

Sample Run 1979

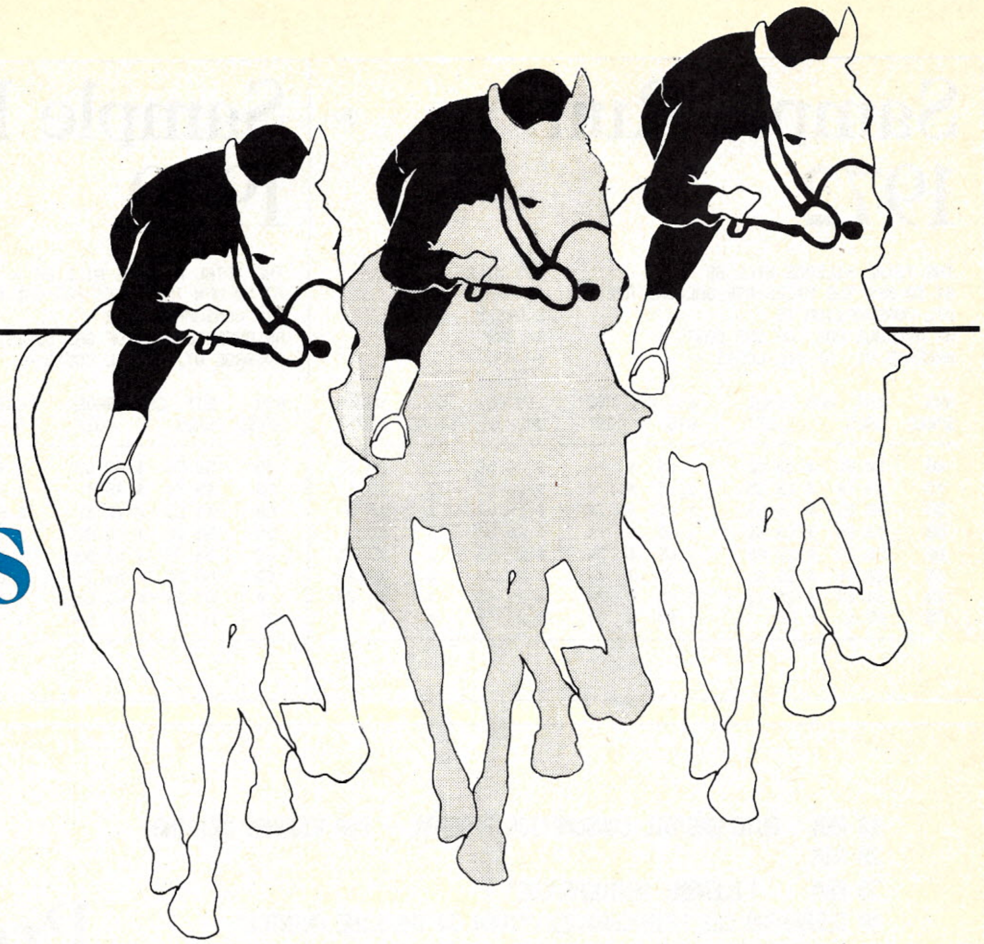
THE TOTAL MILEAGE WILL BE - 825 MILES
AT 50 MPH THE MILES-PER-GALLONS ARE - 17 MPG
OIL CONSUMPTION IS - 2 QTS/M
AN AVERAGE GAL. OF GAS COSTS - $ .979
AVERAGE QT. OF OIL COSTS - $ .95

AVG. SPEED	EST. GAS USAGE GALS.	COST	EST. OIL USE QTS.	COST	TOTAL GAS/OIL	GAS STOPS	HOURS REQ'D.
40	50.96	$ 49.89	1.65	$ 1.57	$ 51.46	3	20.63
45	49.74	$ 48.7	1.65	$ 1.57	$ 50.27	3	18.33
50	48.53	$ 47.51	1.65	$ 1.57	$ 49.08	3	16.5
55	50.96	$ 49.89	1.65	$ 1.57	$ 51.46	3	15
60	56.78	$ 55.59	1.65	$ 1.57	$ 57.16	3	13.75
65	60.66	$ 59.39	1.65	$ 1.57	$ 60.96	4	12.69
70	64.54	$ 63.18	1.65	$ 1.57	$ 64.75	4	11.79
75	69.4	$ 67.94	1.65	$ 1.57	$ 69.51	4	11

Program Listing

```
10 REM   AUTO GAS/OIL CONSUMPTION PROGRAM -- FOR HIGHWAY DRIVING
20 REM
30 REM     FILENAME "AUTOGAS/COS"
40 CLEAR1000:CLS:PRINTCHR$(23);"TURN ON THE LINE PRINTER"
50 INPUT"PRESS ENTER TO CONTINUE -";S
60 CLS
300 PRINT:PRINT"THE QUESTIONS BELOW ARE FOR A HIGHWAY TRIP"
310 PRINT:INPUT"HOW MANY MILES WILL YOU DRIVE -";BA
320 INPUT"WHAT IS YOUR 'MILES PER GALLON ' RATE -";BB
330 INPUT"HOW MANY GALLONS DOES YOUR GAS TANK HOLD -";BC
340 INPUT"HOW MANY QUARTS OF OIL PER 1000 MILES DOES YOUR CAR USE -";BD
350 INPUT"AVERAGE COST PER GALLON - FOR GAS IS -";BE
355 INPUT"AVERAGE COST PER QUART - FOR OIL IS -";BF:PRINT
360 LPRINT"THE TOTAL MILEAGE WILL BE -";TAB(40);BA"MILES"
370 LPRINT"AT 50 MPH THE MILES-PER-GALLONS ARE -";TAB(40);BB"MPG"
380 LPRINT"OIL CONSUMPTION IS -";TAB(40);BD"QTS/M"
382 LPRINT"AN AVERAGE GAL. OF GAS COSTS -";TAB(41);"$";BE
384 LPRINT "AVERAGE QT. OF OIL COSTS -";TAB(41);"$";BF:LPRINT" "
390 LPRINT"AVG. ";TAB(7);"EST. GAS USAGE";TAB(26);"EST. OIL USE";TAB(42);"TOTAL";TAB(51);"GAS";TAB(57);"HOURS"
400 LPRINT"SPEED";TAB(7);"GALS. ";TAB(17);"COST";TAB(26);"QTS. ";TAB(33);"COST";TAB(41);"GAS/OIL";TAB(50);"STOPS";TAB(57);"REQ'D. "
410 LPRINTSTRING$(63,45)
420 PRINT"THE COMPUTER IS CALCULATING - & PRINTING OUT"
430 MH=40
440 DATA 1.05 , 1.025 , 1 , 1.05 , 1.17, 1.25 , 1.33 , 1.43
450 READZX
460 F=(BA/BB)*ZX
470 GB=INT((F+.005)*100)/100
480 Z=BE*GB
495 GC=INT((Z+.005)*100)/100
500 W=(BD*BA)/1000
505 OQ=INT((W+.005)*100)/100
510 Y=OQ*BF
515 OC=INT((Y+.005)*100)/100
520 TC=GC+OC
530 U=GB/BC
540 ST=INT(U)+1
550 W=BA/MH
560 TT=INT((W+.005)*100)/100
630 LPRINTMH;TAB(6);GB;TAB(14);"$"GC;TAB(25);OQ;TAB(32);
    "$"OC;TAB(41);"$"TC;TAB(51);ST;TAB(56);TT
640 MH=MH+5
642 IFMH<=75THENGOTO440
645 LPRINTSTRING$(63,45)
650 FORI=1TO15:LPRINT" ":NEXT
660 END
```

Off to the Races



BY RINALDO F. PRISCO

Tough Tony the Tipster called me several weeks ago. "Why don't youse wise up, kid, and put that computer of yours to work?" he said.

"What do you mean Tony? It is working," I replied.

"Look kid, there ain't no percentage playing dumb games with a dumb machine. Here kid, let me lay it on the line for you."

That conversation was the beginning of a new project for me. Tough Tony told me that he represented a certain party who owned a stable of racing horses. This party wanted to buy a computer to help determine whether his horses would win their races. He needed help since he didn't know a thing about computers or programming. Would I help?

I explained to Tony that, although I was knowledgeable about programming, probabilities and statistics, I knew nothing about handicapping horse races. "No problem, kid; I'll tell you all youse need to know." With that statement, my education in handicapping horse races began. Tony provided me with his first-hand knowledge as well as with every book on handicapping that our local bookstore had — many more than I thought existed. New terms entered my vocabulary: form, class, allowance race, claiming race, maidens, tote board, hedging, pace, overlays, furlongs, ...

Eventually several versions of the program Otter (Off To The Races) were developed. They were meant to be rough attempts to handicap a race using only the entered horse's past performances as published in a typical racing form. Some versions neglect many important factors such as class, pace, weight, jockey and so on, concentrating on speed. Other versions are fairly extensive.

Tony became a daily visitor once the programs were up and running. Eventually, he called me and said, "Don't youse touch or change the last program!" It was picking the first or second horse in each race consistently. Of course, this could have been simply a coincidence. I told Tony that we would need many more tests before we could reach any conclusions regarding the program's ability to handicap a race well enough for us to have any confidence in its predictions.

The Program

Handicapping a horse race is remarkably complex. The most that can be realistically hoped for is that a program give the user an "edge". This edge should be sharpened and honed during the course of the program's development. Payoffs in horse racing are based upon the collective abilities of all players to make judgements. A rational player will have a definite edge over players using hunches, names or pins to make judgements.

Any handicapping program will use numerical quantities to rate various qualities of the entries. Different qualities and numerical rating systems distinguish one program from another. Otter can be easily modified to use any such variation. Several versions have been developed based on one method or another. The more factors considered by a method, the better should be the results. There is a trade-off here. While consideration of more factors will result in better predictions, entering all the required data will result in greater expenditures of time.

This article's version of Otter has the advantage of using a relatively small amount of data. It cuts through all the possibilities, getting down to the fundamental quality of speed.

Past performance charts give speed ratings and variances for each of the recent races of each of the starters. Otter computes their weighted mean

Rinaldo Prisco is an associate professor of mathematics at the State University of New York at Oswego.

Illustration by Dolores Brandow

Program Listing

```
10  REM                  OTTER
20  REM         Weighted Speed Version 2.3
30  REM
40  REM             Rinaldo F. Prisco
50  REM             R.D. #7 Edgebrook
60  REM             Oswego, NY 13126
70  REM
80  REM            All Rights Reserved
90  REM
100 DIM X(15),S(15),C(15),M(15),P(15),L(15),T(15)
110 DIM H$(18),T$(20),R$(2),D$(18),N$(270)
120 INPUT "IS DATA ON DISK? ",Y$:IF Y$(1,1)<>"Y" THEN 190
130 INPUT "WHICH FILE? ",F$
140 OPEN #0,F$:READ #0,T$,D$,R$,F,N
150 READ #0,N$
160 FOR I=1 TO N:READ #0,X(I),S(I),L(I),M(I):NEXT
170 CLOSE #0:FOR I=1 TO N:P(I)=I:NEXT
180 GOTO 520
190 INPUT "Which track? ",T$
200 INPUT "What date? ",D$
210 INPUT "Which race? ",R$
220 INPUT "How many furlongs? ",F
230 INPUT "How many horses entered? ",N
240 !:!
250 FOR I=1 TO N
260 !"Enter name of horse",I,": ",
270 INPUT "",H$:L(I)=LEN(H$):N$((I-1)*18+1)=H$
280 P(I)=I
290 !"Enter Morning Line on ",H$,": ",
300 INPUT "",M(I):PRINT
310 NEXT I
320 FOR I=1 TO N
330 H$=N$((I-1)*18+1,(I-1)*18+L(I))
340 !CHR$(11),TAB(20),H$,"  PP#",P(I)
350 FOR J=1 TO 63:PRINT "*",:NEXT J
360 PRINT:PRINT
370 INPUT "How many prior races? ",M:IF M<11 THEN 390
380 M=8:STOP
390 PRINT:PRINT
400 X=0:S=0
410 FOR J=0 TO M-1:!%2I,J+1,") ",
420 INPUT1 "SPEED FACTOR: ",D:!TAB(30),
430 INPUT "VARIANCE: ",T
440 T=T+D
450 X=X+(M-J)*T:S=S+T*T*(M-J)
460 NEXT J
470 M=M*(M+1)/2
480 S(I)=SQRT((M*S-X*X)/(M*M))
490 X(I)=X/M
500 PRINT
510 NEXT I
520 REM              - SIMULATE RACES -
530 INPUT "ENTER A NUMBER BETWEEN 0 AND 1 ",Z:Z=RND(Z)
540 Q=100
550 FOR I=1 TO N:C(I)=0:NEXT I
560 PRINT #P,"And they are off and running!! ",
570 FOR K=1 TO Q
580 FOR I=1 TO N
590 A=RND(0):Z=4.91*(A^.14-(1-A)^.14)
600 T(I)=X(I)+Z*S(I)
610 NEXT I
620 W=1:M=T(1):FOR I=2 TO N
630 IF T(I)<=M THEN 640:W=I:M=T(I)
640 NEXT I:C(W)=C(W)+1:!W,
650 NEXT K
660 REM        - SORT -
670 FOR I=1 TO N-1:FOR J=I+1 TO N
680 IF C(P(I))>=C(P(J)) THEN 690:T=P(I):P(I)=P(J):P(J)=T
690 NEXT J:NEXT I
700 !:!
710 INPUT "Hard copy? ",Y$:IF Y$(1,1)="Y" THEN P=2
```

continued

(more recent races carry more weight) and standard deviation, providing a speed profile for each starter. These profiles are then used in simulated races. Otter simulates 100 (easily modified to any number) races. Their results are used to estimate the fair odds for each horse winning the actual race. The horses are then sorted according to these odds, with the most favored entry listed first.

Objective

The objective of the program is not to pick the horse with the "best" chance of winning. Rather, Otter searches for the most "underrated" horse in the race. In player parlance, this is called the *best overlay bet*.

Otter computes the best overlay bet by comparing its computed odds with the "morning line," the odds as estimated in various scratch sheets and newspapers. A player at the track would use the tote board to determine the most recent track odds. The largest of the ratios of the track odds to the computed odds is the best overlay bet. In the long run, playing underrated horses should result in larger expected payoffs per investment.

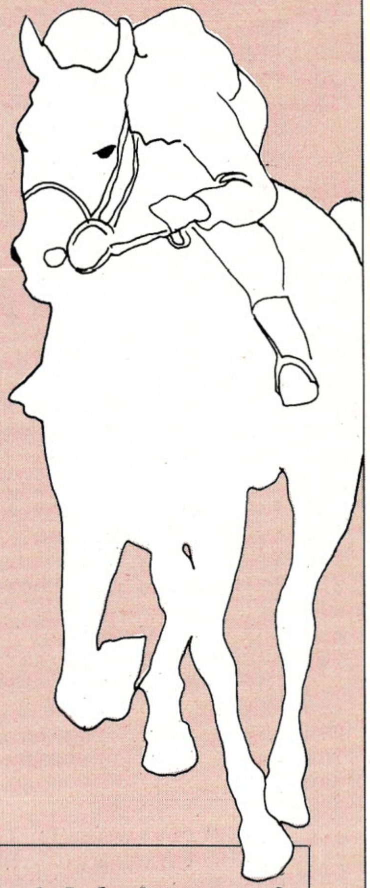
Running the Program

You have the past peformances of the entries in the race before you (available at many newstands). Load Otter, enter RUN and you are off! Once all the data is entered, the simulated races commence. After each is run, the post position of the winner will print out to the screen. (This would be quite striking if we added a graphics routine.)

After several minutes, the results are in. You then await the final printout, almost like being at the track itself. Some typical printouts are included in the Sample Runs. In these examples we used the actual finishing position of each starter in place of their post positions as well as the actual track odds in place of the morning line. This was done to compare the predicted results with the actual results. In some instances we fared pretty well; in others we bombed out. As stated before, all we can hope for is an "edge".

You might be interested to know that Tough Tony and friend now have a microcomputing system of their own using a super version of Otter. Moreover, I think that Tony's ideas about computers have changed. The last time that I saw him the races were over and he was playing Startrek, "rubbing out" Klingons. □

Program Listing continued



```
720 !#P:!#P,"Race #",R$,TAB(25),T$,"    ",D$,
730 !#P,TAB(57),F," furlongs"
740 FOR I=1 TO 69:PRINT #P,"=",:NEXT I:PRINT #P
750 !#P,"    NAME            PP      #WINS          ODDS",
760 !#P,"           MLINE            OL"
770 FOR I=1 TO 69:PRINT #P,"-",:NEXT I:PRINT #2
780 FOR J=1 TO N
790 I=P(J)
800 !#P,N$((I-1)*18+1,I*18),
810 !#P,%3I,I,%9I,C(I),
820 IF C(I)<>0 THEN 850
830 !#P,TAB(39),"----",TAB(50),%6F2,M(I),TAB(65),"----"
840 GOTO 890
850 C(I)=(Q-C(I))/C(I)
860 IF C(I)<>0 THEN O=M(I)/C(I) ELSE O=0
870 IF O<O1 THEN 880:O1=O:B=I
880 !#P,%13F2,C(I),M(I),O
890 NEXT J:!#P
900 !#P,"Best Overlay Bet: ",N$((B-1)*18+1,(B-1)*18+L(B))
910 INPUT "SAVE DATA? ",Y$:IF Y$(1,1)<>"Y" THEN 980
920 INPUT "WHICH FILE? ",F$
930 IF FILE(F$)=-1 THEN CREATE F$,3
940 OPEN #0,F$
950 WRITE #0,T$,D$,R$,F,N,N$
960 FOR I=1 TO N:WRITE #0,X(I),S(I),L(I),M(I)
970 NEXT I:CLOSE #0
980 INPUT "ANOTHER PRINTOUT? ",Y$
990 IF Y$(1,1)="Y" THEN 720
```

Sample Run

Race #2 BELMONT PARK 7/18/79 8.5 furlongs

NAME	PP	#WINS	ODDS	MLINE	OL
All Our Hopes	3	22	3.55	4.90	1.38
Sharmadidit	6	18	4.56	21.50	4.72
Mr. Champ	2	17	4.88	5.00	1.02
Trout Mound	5	14	6.14	5.50	.90
Jungle Bells	4	10	9.00	9.00	1.00
Slow to Anger	1	8	11.50	13.00	1.13
Altar	7	7	13.29	2.10	.16
Artful Pretender	8	4	24.00	14.00	.58
Eleven Shamrocks	9	0	----	11.00	----

Best Overlay Bet: Sharmadidit

Race #3 BELMONT PARK 7/18/79 8.5 furlongs

NAME	PP	#WINS	ODDS	MLINE	OL
Rolling Stone	1	33	2.03	1.90	.94
Prospector's Joy	7	27	2.70	8.00	2.96
Forecast Clear	3	15	5.67	3.10	.55
New Appeal	8	15	5.67	16.60	2.93
Modest Ridge	4	5	19.00	22.70	1.19
Loudly	6	2	49.00	22.20	.45
L'Arsouille	5	1	99.00	9.10	.09
I Celebrate	2	1	99.00	6.70	.07
Think Tomorrow	9	1	99.00	8.60	.09

Best Overlay Bet: Prospector's Joy

How to write for Personal Computing

You've written the programs we want to publish. You – the *Personal Computing* readers – are using your computers in businesses, homes, offices and schools. Other readers, just as software-hungry as you, are eager to try out your programs, your applications and your techniques. So why not share what you've done by submitting an article to *PC?*

It's easier than you might think. Remember: we're more interested in practical programs and useful applications than in fancy prose. And our editorial staff stands ready to help with any problems you encounter in writing your article; just give us a call at (617) 232-5470.

Here are some handy guidelines to help you get started.

First, decide what kind of article you want to write. Do you have a *business program* that will help an executive, salesman, doctor, lawyer or shopkeeper function more efficiently? Think about how businesses can benefit from microcomputers – not only in the obvious areas of inventory, accounting and payroll, but in all departments and levels right up to the president's desk. Financial and marketing analysis, time management, planning, material handling, product design and cost accounting are areas ripe for creative programming.

How do you use your computer for *home and personal applications* in your living room, kitchen, study or den? Again, think beyond the obvious areas of checkbook balancing and budgeting (though these areas are far from exhausted) to other applications. Hobbies, home management, household inventory,, gardening and landscaping, personal income and expense analysis, personal mailing lists and word processing are just a few ideas to spark your imagination.

What *education programs* have you written for children, adults, professionals, businessmen and teachers? Computers can not only teach children basic subjects such as spelling, math, geography, economics, civics, grammar, literature and science, but can help adults review or sharpen skills in these areas as well. How else can computers function in or out of the classroom to aid learning? To help teachers and administrators?

Are you proficient in some programming technique or special computer area you could explain in a *tutorial article*? How do you save time, money, computer memory or frustration when programming or using your computer? Others can benefit from the same techniques you use.

Computer games, history, humor and *fiction* are other areas rich in article and story ideas.

Your second step is to write the text of the article. Remember, readers aren't familiar with your program. So explain in detail what the program does and how it does it. Include here the overall structure of your program as well as any special algorithms or routines you've used. Give suggestions for modifying or expanding the program for other applications, other businesses or other situations.

Third, prepare your supporting documentation. Include at least a program listing and one or two sample runs, and add program notes to explain any special commands used or other special features of your program. Use charts, diagrams, figures and photos if they help explain your program and its use.

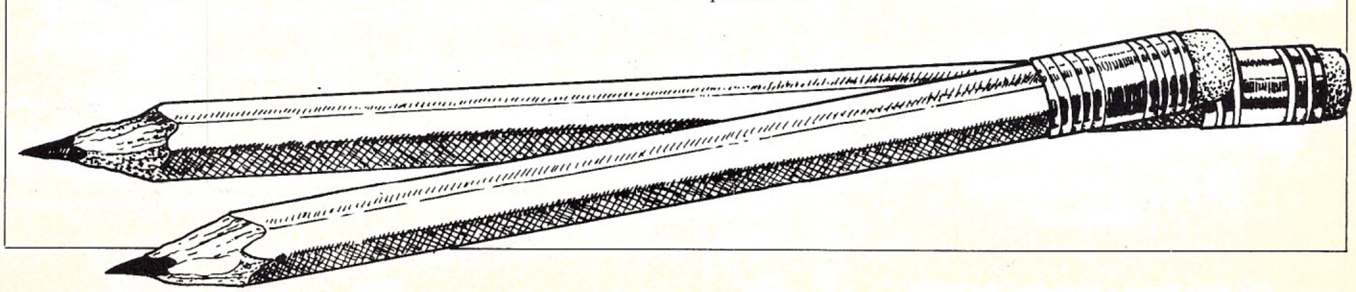
Finally, mail your manuscript. Address it to Editor, Personal Computing Magazine, 1050 Commonwealth Ave., Boston, MA 02215.

A few suggestions: All submissions should be original, typed (*not* all CAPS), double-spaced and neat. Please include your name and address on the first page of the article and enclose a self-addressed, stamped envelope for return of material.

Since we photograph program listings and sample runs exactly as you send them to us for publication in the magazine, please be sure you use a fresh ribbon for computer printouts. If you don't have a printer, you can type your listings single spaced; but again, be sure you use a new ribbon. (If your program relies heavily on graphics, you can photograph sample runs from your CRT. But take care to avoid distortion due to the curve of the screen.)

Feel free to call us if you have any questions or want to discuss specific ideas. We can give you feedback and suggest appropriate slants and approaches.

We're always looking for fresh, original ideas. While these guidelines will help you in preparing material for *Personal Computing*, don't assume we don't want your idea just because it's not mentioned here. Let us and our readers know what *you're* doing with your computer.



HARRY SHERSHOW — Dept. Editor
MORRIS MILLER — Chess Annotater

"Rating Chess Challenger 7"

BY DAVE WELSH

(This month's chess section begins an analysis of a stand-alone chess device (Chess Challenger). The author, David E. Welsh, is a consulting mechanical engineer from Los Angeles, CA 90027, 3124 Rowena Ave. He is a strong chess player (last USCF rating: 1952.) Because, he says, he has found it difficult to swallow the advertising claims from the manufacturers of current chess machines, he decided to evaluate one very thoroughly. He obtained the use of a "representative" machine for several weeks and, during that time, subjected it to an intense scrutiny. Some of his results, including some sharp criticisms plus a few compliments, appear in the following article. Other games of his "tournament" will appear in future PC issues.)

Because the $100.00 barrier has been broken (Chess Challenger 7 lists at $99.95, JS&A's game is under $100.00 and Boris Diplomat is not much more), anyone unwilling to spend $200.00 plus for a "toy" that plays weak chess may want to take another look. I did, and had the luck to borrow a CC 7 for a 10 game "match" in a two-week period. We contested 4 of the games at Level 3 (Average Response Time 1 minute 20 seconds), 4 games at Level 7 (ART 3 minutes), and 2 games at Level 6 (ART 6 minutes). The result was predictable. Welsh: 10 - CC 7:0, though CC did get the better game on two occasions.

CC 7 is at its best at Level 3. Games at Levels 6 and 7 tend to become all-day affairs, and the increase in CC's playing strength is by no means proportional to the extra time used. The package could be improved (the board for example, atop CC is really too small). If CC must supply a coded board, why not a separate folding one?

How strong is CC 7? Not very. It is probably misleading to suggest a numerical rating, but most players above 1200 can usually defeat it. CC will give beginners and casual players who don't know "book" a battle. However, it is a good opponent for a youngster learning the game. More advanced players will soon discover its weak points, and then tend to "play the opponent" rather than the board. Instead of presenting a general description of CC's playing peculiarities, I offer them as I encountered them — in the games.

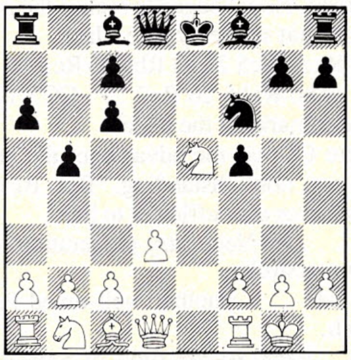


Diagram 1

GAME 1 - LEVEL 3

This was not much of a contest, and doesn't show CC to advantage.

BLACK: CC7 Ruy Lopez 1 e4 e5 2 Nf3 Nc6 3 Bb5 a6 4 Ba4 Nf6 5 **o-o** b5 6 Bb3 Nxe4?! So far CC has played a reasonable line, but this is dangerous. 7 Bd5 . . . By this simple move, White gets a clear advantage. 7 . . . f5? a dangerous weakening of CC's position. 8 Bxc6 dxc6 9 d3 Nf6 10 Nxe5 (See Diagram 1). . .

At this point CC is busted.

10 . . . Qd5? Tactically unsound; CC has nothing better than 10 . . . Bb7. 11 Nc3! Qxe5 12 Rel Ne4 13 dxe4 Now White has the piece back and Black is unable to develop.

13 . . . Be6 14 exf5 Qxf5 15 Qe2

15 . . . Kd7? Correct was . . . Kf7.

16 Bf4! b4? 17 Rdl+ Ke7 18 Bg5+ Ke8?

Again . . . Kf7 was much better.

19 Ne4 h6 20 Bh4 Kf7 CC was wise to avoid 20 . . . g5? 21 Qh5+ Qf7 (21 . . . Ke7? 22 Nxg5!; 21 . . . Bf7?? 22 Nf6 mate) 22 Nf6+ Ke7 23 Rd7+ Kxf6 24 Rxf7+ Bxf7 25 Qf3+ and wins.

21 Bg3 Ra7 22 b3 Be7 23 Rd3 . . .

23 . . . Rb7?? I still can't understand how CC could have overlooked this! . . The remaining moves were: 24 Rf3 g6 25 Rxf5+ Bxf5 26 Qc4+ Kg7? 27 Be5+ Kf8 28 Bxh8 Bd7 29 Nc5 Bc8? 30 Ne6+ Bxe6 31 Qxe6 Rb5 32 Qxe7+ 1 - 0

GAME 2 - LEVEL 3

This game was more interesting; although CC again got in trouble in the opening, its decisive error was not quite so obvious, and the game continued to an ending.

WHITE: CC 7 4 Knights' Game 1 e4 e5 2 Nf3 Nc6 3 Bc4 Nf6 4 Nc3 Bc5 CC didn't know the 2 Knight's Defence and transposed into the Italian variation of the 4 Knights' Game. Here 4 . . . Nxe4 is better for Black, but I wanted to see what CC would do in a quiet opening. 5 d3 d6 6 Be3 Bb6 7 d4 . . .

This is premature.

7 . . . exd4 8 Bxd4 . . . Normal (for humans) would be 8 Nxd4.

8 . . . Bg4! Suddenly White is under inconvenient pressure.

9 Qd3? . . . Overlooking a combination; necessary was 9 Bxb6 axb6 followed by 10 Be2.

9 . . . Bxf3 10 Bxf6 Qxf6 11 Qxf3

11 . . . Qxf3 12 gxf3 Nd4

CC now loses a Pawn.

13 0-0-0 Nxf3 14 Rdfl?! . . .

The f-pawn wasn't really en prise. 14 . . . Ne5 15 Bb5+ c6 16 Be2 0-0-0 17 Rhgl g6 18 h4 Bd4 19 Rg3 Nd7 20 Bc4 Rhf8 21 f4 Nc5

CC's "plan" is quite harmless — mine isn't. Now it blunders. (Diagram 2)

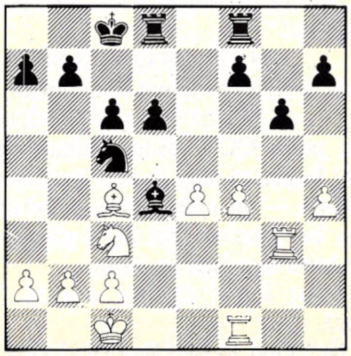


Diagram 2

22 Rd1? . . . Obviously CC is only analyzing to a depth of 3 half-moves. This loses another Pawn and the game.
22 . . .Bxc3 23 Rxc3 Nxe4 24 Rb3 d5 25 Be2 Rfe8 26 Rb4?! Kc7! 27 Bg4 . . . CC sees the threat of Nc3!
27 . . . f5 28 Bf3 Nf6 29 Ra4 Re3 This invasion is decisive.
30 Bg2 a6 31 Rad4 Rde8 32 a4 Re2 33 Bf3? . . . Much better was 33 R4d2.
33 . . . Rh2 34 Rb4 Rxh4 35 Rd3 a5 36 Rbd4 Ne4 Now it's all over.
37 b4 . . . CC will pursue a "plan" to the death, even a bad one.
37 . . . Rxf4 38 bxa5 Ra8 39 Rb4 Rxa5 40 Rdb3 b5! The price of the Pawn is exchanges.
41 axb5 cxb5 42 Bxe4 dxe4 43 Rxb5 43 . . . Rxb5 44 Rxb5 h5
Now Black has an easy win.
45 Kd2 Rf3 46 Rc5+ Kd6 47 Rc4 h4 48 Kd1 h3 49 Rd4+ Kc5 50 Rd2 Rg3 51 Kc1? Rg2 52 c3 Rxd2 53 Kxd2 h2 54 c4 h1(Q) 55 Kc3 Qd1 56 Kb2 Kb4 57 c5 Qd2+ 58 Ka1 Kb3
0-1

GAME 3 - LEVEL 3

CC rebounds with its best effort of the match, aided by careless play on my part. It held the balance up to move 53 before suddenly collapsing.
WHITE: CC 7 Queen's Pawn Opening 1 d4 Nf6 2 Nc3 d5 3 Bf4 Bf5 4 e3 e6 5 Be2 . . . CC plays very conservatively in the opening; more aggressive is 5 Bd3 c6 6 Ne2 Nd7 7 Ng3 Bxd3.
5 . . . c5 6 Nf3 Nc6 7 o-o Be7 8 a4 . . . A reasonable move in this position.
8 . . . o-o 9 Ne5 Nb4 Time to get some action going; this game is becoming terribly boring.
10 Bd3?! . . . Obviously better is 10 Rc1.
10 . . . Nxd3 11 cxd3? . . . Apparently CC thinks its Knight is too well posted to move!
11 . . . cxd4 12 exd4 Qb6 13 Nb5?! . . . Loss of a Pawn cannot be avoided.
13 . . . a6 14 Nc3 Qxd4 15 Qf3 Bb4 At this point CC's all-too-human opponent began to play in very perfunctory fashion — in extenuation of which it may be observed that CC's play so far has been less than inspiring.
16 Rfb1 Rac8? Correct was 16 . . . Bxc3.
17 Ne2 Qb6 18 Be3 Bc5 19 a5 Qb4
20 Qf4 Qxf4 21 Bxf4 Ba7
Now it looks as if White must play Rc1 with a fine ending for Black.
22 Ra4!! . . . A really excellent move, which I completely failed to appreciate even after it was played.
22 . . . Rc2? 23 Nd4 Bxd4 Clearly forced. Now I began to realize the problems that suddenly face Black.
24 Rxd4 Rc5 25 b4 Rb5 26 Rc1 b6
27 axb6 Rxb6 28 Rc7 . . . CC's high-water mark for the match.
Here CC has the advantage, the Pawn minus notwithstanding. But Black's resources are sufficient to hold.
28 . . . h6 Necessary preparation for a counterstroke.
29 h4! . . . Another well-timed Pawn push.
29. . .Rfb8! By this time, of course, my interest in the game has become intense — there's a good deal more to this move than meets the eye.
30 Bd2! . . . Did CS see 30 Rxf7? Rxb4 31 Rxb4 Rxb4 32 Bg3 Rb1+ 33 Kh2 Ng5+ etc. — if not, why didn't it want to trade the weak b-Pawn for my f-Pawn?
30 . . . R6b7 31 Rc6 Ra8 Now Black is out of danger and can work on freeing his position.
32 f3 Ne8 33 Ng4 . . . The motive for this is unclear; I was glad to avoid opposite-colored Bishops, by . . . 33 . . . Bxg4
34 fxg4? . . . Really strange. Of course 34 Rxg4 was much better, but perhaps CC thought its Rook ideally posted at d4!
34 . . . Nc7 35 Rf4 e5 36 Rf3 Ne6 37 Rd6 d4 38 Kf2 Nf8 39 h5 f6
Now Black gets his King into the game.
40 Ke2 Kf7 41 Rc6 Ke7 42 Rf2 Kd7 43 Rc1 Ne6 44 Rf5 Nc7 45 g3 Nd5 46 Rc4 Rab8
At this point it seems that the Pawn must fall.
47 Rf1! . . . Now if Black takes the b-Pawn, CC wins the a-Pawn in return.
47 . . . Nxb4?! . . . Here I thought my a-Pawn was the weaker of the two; correct enough; but 47 . . . Nc3+! trading the d-Pawn for White's b-Pawn was much better. Now the ending ought to be very hard to win.
48 Ra1 Kd6 49 Bxb4+ Rxd4 50 Rxa6+ R4b6 51 Rc8 . . . Tricky, but this just exchanges a pair of Rooks.
51 . . . Rxc8 52 Rxb6+ Rc6 53 Rxc6+? CC commits suicide in the ending. After 53 Rb8, Black would be faced with extreme technical problems in attempting to win the ending, due to the vulnerability of the g-Pawn.
53 . . . Kxc6 54 Ke1?? . . . The last hope, faint though it is, is 54 Kf2 with hopes for 54 . . Kd5 55 Kf3 Ke6?
56 Ke4 Kd6 57 Kf5; which I would have short-circuited by 55 . . . e4+!
54 . . . Kd5 55 Kd1? e4 56 dxe4+ Kxe4
CC was now playing very rapidly, about 10 seconds a move.
57 Ke1? Ke3 58 Kf1? d3 59 g5 d2 60 gxf6 d1(Q)+ 61 Kg2 Qe2+ 62 Kh3 Kf3 63 fxg7 Qg2+
0 - 1

An 1800 Level Chess Game

Based in part on my experience with CC 7, I've formulated some ideas on what will be required for an electronic chess game to play at approximately 1800 level:

1) Look-ahead capability — 4 moves @ 4 minute response time (approximately international tournament time rate). This requires (in an open position) an evaluation rate of about 30,000 positions/sec. — about 1000 times CC 7's capacity. Given the present rate of progress in electronics, this may be the easiest requirement.

2) Positional evaluation — A sophisticated positional evaluation program is a must. Even today's big-machine programs are relatively weak in strategic insight; they play better in open, tactical positions. Briefly, a good positional evaluation program must correctly appreciate strong and weak squares; must value the control or occupation of each square according to the position at the moment as well as probable future positions, instead of using fixed values all through the game; must be able to appreciate the potential activity of modern closed positions (a particularly difficult task); and must be able to foresee and initiate strategic piece maneuvers, for example the well-known Nd2-f1-e3-f5 maneuver in the Ruy Lopez. All this will be very difficult in practice.

3) Endgames — A reasonable level of endgame proficiency should not be too difficult to attain, exact analysis being much easier in simplified positions. A different program, however, is required for the endgame — and it must

be possible to switch to this program in middle-game analysis when endgame positions must be evaluated, a difficult thing to do in practice.

4) Openings — Big-machine program experience indicates that a "book" of several thousand variations is required for competitive survival. This is largely due to the difficulty of writing a good openings program. Even if this were not impractical, an electronic game would surely be more interesting and valuable to serious players if it followed the best theoretical lines, rather than making second-best though reasonable moves. Obviously some kind of mass memory is essential, and it is not easy to visualize how enough memory for a large openings book could be supplied. Here the game manufacturer should profit from the example of calculator manufacturers, by supplying the openings library in the form of replaceable modules — one general-purpose module free with the machine, and optional specialized modules (e.g. an entire module just for the Ruy Lopez) sold separately to serious players. It would also be wise to make the modules re-programmable.

If all this can be done for less than \$300.00 in today's money, not only will the game manufacturer have a highly successful product with ongoing revenue from module sales, but Chess in the U.S. will benefit greatly from mass exposure to the game at a relatively high standard of play.

Other games of Welsh's 10-game "tournament" will appear in following issues.

General Ideas....

Words from Russia

Some of the articles on computer chess which appear here are occasional reprints from the Russian, pocket-sized digest called SPUTNIK. The magazine is a colorful presentation of life and activities (including chess news) in Russia and contains a choice number of articles (30 to 40) extracted from all the publications in the Soviet Press. Annual subscriptions to this English language version of what they call a "monthly digest of the best current Soviet writing"can be obtained for \$7.50. A list of subscription agencies in the USA and in 14 other foreign countries can be obtained by writing to SPUTNIK DIGEST; Novosti Press Agency; 2 Pushkin Square; 103006, Moscow, USSR.

Beyond comprehension?

"In the September issue of PC," writes John Hesse, of 6583 Hollandaire Dr. W., Boca Raton, FL, 33433, "a Russian writer indicates the complexity of solving the game of chess by computer. To reproduce the first ten moves alone would 'require the world population and 217 billion years,' he says. And he states that all possible combinations on the board approach 10^{120}. Can this number of moves ever be duplicated by a computer?

"The answer is no, it cannot. The reason derives from fundamental physical limits which may be interesting to some of your readers. The limits were worked out by the mathematician Hans J. Bremermann writing on 'Complexity and Transcomputability' in THE ENCYCLOPEDIA OF IGNORANCE, edited by Duncan and Weston-Smith and published by Simon and Schuster, NY, 1977. These limits are set by:

1. The age and size of the universe.
2. The speed of light.
3. Quantum mechanics.

"The first item determines the maximum size and time limit of our ultimate computer: the entire known universe is used in the construction of the chess-playing system. The other two items, as determined by Bremermann, establish a maximum internal signal flow rate. The speed of light is the maximum speed at which information can flow from point to point. Quantum mechanics tells us the minimum amount of energy required to reliably represent one bit of data.

"The maximum rate is c^2/h where c is the speed of light and h is Planck's constant. This computes to 1.35×10^{47} bits per second per gram, and assumes the *entire* mass of the system is in the form of data energy (in the form of photons, for example). This is clearly an impossibility because it is difficult to imagine a system that has some kind of long-term structure but also has a zero rest mass! Even a hydrogen bomb or an exploding super-nova has energy flows per unit of mass many orders of magnitude less than the above figure. Note that current computer systems achieve about 10^5 bits per second per gram. Nevertheless, this is the limit for information processing.

"The estimated age of the universe is 20 billion years or 6.3×10^{17} seconds. Its mass is about 10^{55} grams. The mass-time product then is 6.3×10^{72} gram seconds. Therefore, the maximum number of states this computer could achieve is $6.3 \times 10^{72} \times 1.35 \times 10^{47}$ or 8.5×10^{119} — slightly less than the 10^{120} possible chess board combinations mentioned initially (less by about 3 billion years!).

"If the Big Bang that created the cycle of the universe that we live in had generated, instead, a large, fast, chess-playing computer, it would still be playing today.

"In this example, the entire universe is one big CPU with no mass set aside for memory. Interesting games and other results cannot be preserved. Perhaps more interesting than the sheer unsolvability of the chess problem is the giant gap that exists between current DP technology and the physical limit: about 42 orders of magnitude!"

We're waiting

The following letter from Al Gallia, 15615 Profit Ave., Baton Rouge, LA 70816, reveals the current dilemma among "computer-oriented" chess players. "I am interested in purchasing a micro chess unit," writes Al, "but I don't have enough information on which to make a selection. Being a 1500-rated player I would prefer a unit that could play above my level. At least, it should be able to play a relatively even game to my playing level,

otherwise I wouldn't consider it worth buying. I am willing to spend up to $300 or so. I eagerly look for all information on the current commerical stand-alone units. I am especially interested in relative speed and program complexity. So, where are these machines?"

It is apparent from Al's letter, which is similar to many others we receive, that there is a waiting market for chess devices which will play at higher levels than current estimations. Watching the development of both chess programs and the availability of new bubble--memory hardware, one must arrive at the conclusion that manufacturers of devices like Chess Challenger, Boris, and others cannot be too far away from the magic level of 1500. Once such a rating has been clearly established for micro chess programs (casettes and disks which run on a microcomputer already are at that plateau) we feel that the market will open up.

Hail to the champ!

That is the huzzah being heard from Hayden Book Company these days, in the following press release:

"Here's the program, the best of the micros, that came in third against the big machines (mainframes and maxis) at the 9th North American Computer Chess Championship! Published by Hayden Book Company, Inc. it is entitled SARGON II by Dan and Kathe Spracklen. 'Buy this program when it becomes available . . . unequaled in the end game' . . .*Personal Computing,* July, 1979. SARGON II is able to push passed pawns toward queening, play a strong end game, and range in deep play levels at end game without user direction. The computer displays the levels of play at which it is thinking, and also shows the move it is currently thinking of making. And by a flashing asterisk, it shows that the computer is thinking and has not accidentally hung up. The computer will keep changing the move shown until its final choice is made. SARGON II has 7 levels of play, and levels 0-3 play in tournament time. It has a randomized opening book for all 7 levels of play through 3 moves. When setting up the board, the user can scan up and down, left and right. And finally, for those players who may need help, a special hint mode is included at all levels of play but 0. When you consider the features and play capabilities of SARGON II, it's easy to see why it's the Champ of Champs!" (We have heard mutterings from the Chess Challenger people saying: "Well, we'll just have to see about that! Champ of Champs! Indeed!")

Flash points

As the Jan. PC was about to go to press (Oct. 31) word was received that CHESS 4.7 had won the 10th ACM Computer Chess Tournament in Detroit. SARGON/BORIS finished in seventh place. MYCHESS finished in fifth place . . . A cable message received at the same time said that the Boris/Sargon unit had won the first four rounds in David Levy's European Tournament in London, ahead of both MYCHESS and CHESS CHALLENGER . . . More complete details will appear next month . . . The U.S. Chess Federation is co-sponsoring the Russian Gambit Trip this year. The tour takes Americans to famous Russian chess centers, including computer-chess labs, and meets with world-renowned chess leaders. Cost of the all-expense-paid trip (with meals) has been reduced to $1495. For further information, contact Malcom Byrne, Citizen Exchange, 145 Hanover St., Boston, Mass. . . . An energetic chess addict, Varn Fields, of Philadelphia, has been an active postal chess participant for many years. He is now launching a postal chess tournament for computers. He thinks such a tournament may last about a year and he will be reporting regularly on the progress in the contest. Anyone with a computer program (their own or commercial), or with a stand-alone device, who would like to participate in this tournament, should respond to Varn's ad in the classified section.

A Mauling at the Mall

The above play on words is an accurate description, according to John Urwin, of what happened one day last October when a small microcomputer chess-program (MYCHESS, on a CROMEMCO) challenged a mainframe-program (XENARBOR-4, on a Control Data) to a game of chess. All the dominated, struggling, hard-working little people of the world will be glad to know that the small skinny program beat the big muscle-bound bully! And hooray, for our side! John Urwin describes the significant event:

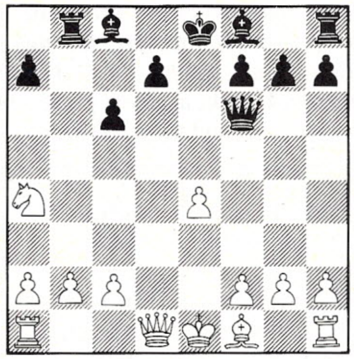
"The demonstration, hosted by The Santa Clara Chess Club, was called The Third Annual Challenge The Masters and was held at the Vallco Shopping Center in Cupertino, CA on Oct. 6th, 1979. It was staged in the open spaces of the mall with boards set up in usual long-table fashion for human play and the computers set up nearby, side by side.

"Three well-known chess masters participated. They were International Grand Master Peter Biyiasas; National Master George Kane; and Life Master Dennis Fritzinger. The three played simultaneously against 15 opponents each. The computers, meanwhile, which were set up for blitz chess demonstration against humans, included a modem connection to a CDC (for Xenarbor-4); a CROMEMCO (for Mychess) and an ATARI machine (for the Atari chess cartridge.)

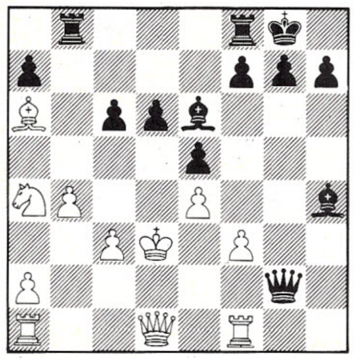
"The masters had a full day and played continuously from 11:00 A.M. to 5:00 P.M. without any time out for a break. Really, a Trojan performance! But the spectators (and the estimate was that by the end of the day, several thousand had milled around the demonstration) were more fascinated by the computers than by the chess masters. The average spectator watched the humans play chess for only about 10 or 15 minutes after which they became bored with the relative inactivity there and came over to the machines. The young

chess addicts, of course, followed the grandmasters worshipfully, jotting down all the moves and making occasional outcries of approval. But the real interest was at the machines. Anyone who doubts that 'machine chess' has already spawned a deep interest among the country's chess players should have been at the Vallco Shopping Center. The majority of the spectators spent most of their time here. They examined the units; touched the keyboards as though they were fingering the textures of fine cloth and they asked many questions. How did the machines work? How were moves selected? How strong were their chess playing abilities? How much did they cost? Would they break down after so many games? All of the questions were answered and most of the spectators became instant potential buyers of the commercial devices. When they would actually break down and buy a unit, they would not say. But from this shopping mall experience, I would guess that the market for chess devices is growing stronger all the time and it is only a question of time before chess players break open their penny banks to buy one of the machines.

"An interesting moment of the machine demonstration arose when MYCHESS, the little guy, challenged giant XENARBOR 4 to a 'fight'. Some



(Diagram A)



(Diagram B)

words were exchanged before the match began. The operator for the CDC computer sneered at such a match, saying it was rather ridiculous and that the giant computer would have no trouble trouncing the little upstart. But plucky MYCHESS persisted in its challenge and finally, XENARBOR said okay. So, it was agreed to play one game.

"We began our game with XENARBOR 4 playing white and MYCHESS playing black at a 4-ply level. The first scheduled game ended with a draw by repetition much to XENARBOR's dismay. The second game was a different story. At a point where underdog MYCHESS was ahead by three minor pieces and was well on its way toward winning the game, XENARBOR entered a wrong move! It was decided not to waste time by resetting the positions even though XENARBOR was in trouble, but to play a third game.

"On this final game, MYCHESS appeared to be doing poorly with four pawns already down! But on move 41, my program showed its bullhorns and found a forced mate! It was truly an historical event with the puny microcomputer pretty well dominating the big mainframe. I am toying with the idea of now calling my program 'MYCHESS THE GIANT KILLER'.

"Meanwhile the Atari chess machine was tied up most of the day by an expert-rated player, (Sanchez, 2160) who was so fascinated by computer chess that he could not pull himself away. He

Notes to game Mychess vs Xenarbor: by Morris Miller:

(a) Book, but I prefer 6-N-KB3, B-N5; 7-B-QB4, 0-0 [not 7-. . .NxP; 8-BxPch, K-B; 9-0-0!, NxN; 10-PxN, BxP; 11-B-R3ch, KxB; 12-Q-Q5ch, K-K1; 13-KR-K1! with a mating attack; or 12-. . .K-B3 or N3; 13-QR-N1 followed by R-N3 and the rook goes over to the king side] 8-0-0!, BxN; 9-PxB, NxP; 10-R-K1, NxQBP; 11-Q-Q3, N-R5; 12-B-R3, R-K1; 13-B-Q6. White will get back one of his two pawns and have the two bishops and a crushing advantage. The point of the above variations, which are of course far beyond the lookahead of the programs, is that the input of mere opening moves is not enough. The principles behind the moves have to be instilled or disaster follows.

(b) Best is 9-R-QN1. Mychess is afraid of the pin and indeed seems to value knights over bishops, as evident by its previous move, a concept that is bad and should be eliminated. Black now has the move R-N5, winning a pawn at K4 or N7.

(c) Starting a wholly unwarranted king side series of moves. K-R1 is best.

(d) Now the knight is marooned.

(e) White starts throwing away pawns needlessly, which black seems strangely reluctant to take.

(f) Instead P-KB4 followed by P-Q4 smashes through the center.

(g) Now instead of the text, which succeeds in stalemating the queen, there is a mating finish: 27-. . .P-Q4ch!; 28-BxP, PxBch; 29-Kxany pawn and the rooks come on the open files with a quick mate. Or 27-. . .P-Q4ch; 28-K-Q3, P-K5 mate. It is perfectly all right to sacrifice pawns to mate!

(h) At last a threat! Can anyone believe white will win? But in the last half of the game black plays listlessly and white vigorously.

(i) Missing a chance for a different win: 35-R-KN1 and if B-N3 as in the text; 36-RxPch, KxR; 37-Q-Q7ch, K-N3; 38-Q-B5ch, K-N2; 39-R-KR1, R-KR1; 40-Q-Q7ch, K-B1; 41-RxR mate.

(j) R-KR1 forces mate quicker.

David beat Goliath; big is not enough against a better program.

later commented that ATARI made very reasonable moves in its chess games. I find it rather significant that such a good player would play that long with a computer program. My own observation is that as chess programs get stronger, chess interest among the 'purists' is going to balloon. Someday, I think, all chess players will be walking around the street with small battery-operated chess devices in their pockets and kids will be playing computer chess on every street corner, in public transportation or at disco parties, during rest periods by the orchestra."

WHITE: MYCHESS (on Cromemco)
BLACK: XENARBOR 4 (on Control Data Computer)

1.	P-K4	P-QB4
2.	N-KB3	N-QB3
3.	P-Q4	PxP
4.	NxP	N-B3
5.	QN-B3	P-K4
6.	NxN (a)	NPxN
7.	QB-N5	R-QN1!
8.	BxN?	QxB
9.	N-R4 (b)	
	(See Diagram A)	
9.	...	B-K2
10.	B-Q3	0-0
11.	0-0	Q-N3
12.	P-KB3 (c)	P-Q3
13.	K-B2	B-K3
14.	P-B3 (d)	Q-R4
15.	P-QN4 (e)	Q-R5ch
16.	K-K2	QxRP
17.	K-B2	B-R5ch
18.	K-K3	Q-B5ch
19.	K-K2	Q-R7
20.	B-R6	QxPch
21.	K-Q3	
	(See Diagram B)	
21.	...	B-B1 (f)
22.	B-B4	B-R6
23.	R-KN1	Q-KB7
24.	R-QN1	B-N7
25.	P-R3	BxP
26.	Q-KB1	BxPch
27.	KxB (g)	Q-QB7ch
28.	K-K3	QxN
29.	Q-N2 (h)	P-N4
30.	B-N3	QxRP
31.	Q-K4	K-R1
32.	QxBP	R(N1)−B1
33.	QxP	P-B3
34.	K-K4	B-B7
35.	R(KN)−Q1 (i)	B-N3
36.	Q-K7	P-QR4
37.	B-B7 (j)	RxB
38.	QxB	P-B4ch
39.	KxBP	B-K6
40.	R-Q7	QxP
41.	QxP mate	

Origins of Computer Chess

Sometime ago, Dr. I. J. Good, of Virginia Polytechnic Institute and State University, sent us a note describing an unusual game of garden chess involving mental arithmetic which was once conducted by Dr. Champernowne of England against his famous colleague, Dr. A. M. Turing. Also mentioned in the note was a reference to an early chess machine that the two were discussing. For more information on this early experimentation in chess and chess activities we contacted Dr. David Champernowne (Professor Emeritus, Trinity College) and received the following response:

"It is not really correct to describe as 'Garden Chess' the games which Alan Turing and I played in the garden of my mother's house some time near the end of the second world war. They were played at a time when Turing, who was acquainted with Bannister, was taking up long distance running and having considerable success with it. He was interested in the disastrous effects which running had on one's mental capacity to make rational decisions concerning the tactics of the race. Our games were simple experiments to test the effects of running on clear thinking. The two players had to chase one another round a short course and perform a simple mental task at a certain point on the circuit; in one version of the game the task was to double a number (which of course got horribly large after several circuits). In another version the task was to play a move in a chess game. One tactic of that chess experiment was to run very fast and get two moves running, but the after-effects might prove disastrous to one's play. These experiments provided more entertainment than useful knowledge. The game did not combine chess and mental arithmetic. One game involved chess and the other involved mental arithmetic.

"It was in the late summer of 1948 that Turing and I did try out a loose system of rules for deciding on the next move in a chess game which we thought could be fairly easily programmed for a computer. My long-suffering wife, a beginner at chess, took on the system and lost. Neither Turing nor I took this experiment at all seriously and I certainly kept no record of the system and I doubt whether he did either. Here is what I think I remember about the system but I may have been influenced by what I have since read about others people's systems.

"There was a system for estimating the effects of any move on White's estimated net advantage over Black. This allowed for:

(1) Captures, using the conventional scale of 10 for pawn, 30 for knight or bishop, 50 for rook, 100 for queen and something huge, say 5000, for king.
(2) Change in mobility; i.e., change in the number of squares to which any piece or pawn could immediately move legitimately (1 each)
(3) Special incentives for (a) Castling (3 points). (b) Advancing a passed pawn (1 or 2 points). (c) Getting a rook onto the seventh rank (4 points perhaps).
(4) I don't think occupation of one of the 4 central squares gained any special bonus. We did not cater to the end-game.

"Most of our attention went to deciding which moves were to be followed up. My memory about this is infuriatingly weak. Captures had to be followed up at least to the point where no further capture was immediately possible. Checks and forcing moves had to be followed further. We were particularly keen on the idea that whereas certain moves would be scorned as pointless and pursued no further others would be followed quite a long way down certain paths. In the actual experiment I suspect we were a bit slapdash about all this and must have made a number of slips since the arithmetic was extremely tedious with pencil and paper. Our general conclusion was that a computer should be fairly easy to program to play a game of chess against a beginner and stand a fair chance of winning or at least reaching a winning position.

"Both Turing and I were weak players but he was the weaker. We also tried out a poker-playing system based on zero-sum two-person game theory. Whereas I had contributed quite a lot to the design of the chess system I made no contribution to the poker-system except to act as guinea-pig in which capacity I gained enormous satisfaction in defeating the system by good luck.

"Oddly enough your note reached me just when Shawn Wylie (who was mentioned in the Turing letter sent along), now a near neighbour, and I had clubbed together with 3 others to invest in a 'Boris Diplomat'. We are recently retired and can afford time to experiment with this toy."

A brief background on Dr. Champernowne (not updated):

D.G. CHAMPERNOWNE was Reader in Economics at the University of Cambridge and College Lecturer and Fellow (Economics) in Trinity College, Cambridge. Educated at Winchester College and at Kings College, Cambridge, he was assistant lecturer at the London School of Economics from 1936 to 1938 and University Lecturer in Statistics in Cambridge University from 1938 to 1939. From 1939 to 1941, he served as a member of the Prime Minister's Statistical Section, and from 1941 to 1945 he was assistant director of programs: M.A.P. In 1945 Mr. Champernowne was named Director of the Oxford Institute of Statistics and All Souls Reader in Statistics. From 1949 to 1959, he was Professor of Statistics in the University of Oxford. His other honors include Prize Fellow of Kings College, Cambridge (1937-1948) and Fellow of Nuffield College, Oxford (1945-1959).

Currently retired, Dr. Champernowne holds the title of Professor Emeritus, Trinity College, England. He lives at 25 Worts Causeway; Cambridge CB1 4RJ, England.

Dr. Good also has an impressive background. His most interesting post (to readers of spy thrillers) was his work during World War II at Bletchley on "Ultra". (Dr. Champernowne worked on the same project.) This project was the subject of a best selling book (The Ultra Secret) several years ago which described the intense research done by the "best brains of the Allies" in developing a wartime codebreaking machine.

Further achievements marking Dr. Good's distinguished career include: Lecturer, Researcher and Consultant on such subjects as mathematics, electronic computing, information retrieval, evaluation of the Perceptron, statistical computation, probability determinations and other subjects too numerous to list here. Dr. Good, who once played chess at the championship level in England, holds numerous degrees and honors including a B.A., M.A. and Ph.D. from Cambridge plus Doctor of Science degrees from both Cambridge and Oxford. In 1942 he was among seven people in England who helped design a large-scale (classified) binary electronic digital computer. "I believed," he comments, "that this was the first such unit in the world until I learned in 1974 that Atasonoff had told Mauchly how to build a computer in 1938. The two principal designers of our computer were M.H.A. Newman and T.H. Flowers. Flowers headed the engineering group among whom I believe the most influential were S. W. Broadhurst, W.W. Chandler and A.W.M. Coombs. The main users were Newman, Michie and myself." Currently Dr. Good serves as "University Distinguished Professor of Statistics" at Virginia Polytechnic Institute and State University, Blacksburg, VA.

Animated Chess

Chess has not always been a slow, hushed, static activity confined to a table top while players rested their thoughtful, serious faces in upturned hands. One well-known variation of chess which was full of music and movement and gaity was called "Living Chess". A delightful description of "Living Chess" appears in the *Encyclopedia of Chess,* edited by Anne Sunnucks, published and copyrighted in 1970 by St. Martin's Press and is attributed, in the encyclopedia, to E. Pritchard. The encyclopedic item, which follows, is reprinted with permission of the publisher, St. Martin's Press, Inc., 175 Fifth Avenue, New York, NY 10010:

Living Chess

During the past 600 years the game played with living pieces has from time to time been presented in different countries. The spectacle has tended to follow a fairly standard pattern. Contests take place usually in the open air. The chessmen, quite often children of appropriate size, are attractively costumed in contrasting colours. They are naturally taught enough chess to interpret the orders of the two players correctly. The players are seated on something like tennis umpire's ladders, from which they can view the large board and call out their moves, through a megaphone if necessary. In order to keep interest and to avoid tiring the participants, games are usually lightning ones, with each player allowed 10 minutes, as recorded by his own timekeeper. An umpire ensures that the correct moves are made. There are sometimes additional conventions, such as the King lifting his sceptre when in check.

The fifteenth century seems to provide the earliest records of the living game. The Sultan Mohammed is said to have played it in Grenada in 1408. The most famous of all living games is the one still played regularly in Marostica, Italy. In 1454 two suitors played for the hand of the beautiful Lionara. They played out their contest on the piazza for all to see, and it is this same game

which has been faithfully reenacted through the years. The pieces, magnificently attired according to tradition are well rehearsed. The board is enormous, giving scope to mounted Knights and Rooks manned by two soldiers apiece. A herald announces the moves to the audience.

Later, in the sixteenth century, Don John of Austria is said to have had a chamber in which was a chequered pavement of black and white marble. Upon this living men moved under his direction. About the same time we find Rabelais describing in *Pantagruel* (1564) an elaborate living game, into which music and dancing are introduced. The 32 young persons are clad in silver and gold and play and dance on squares of white and yellow.

In more recent years in England displays have largely been organised by the Kent County Chess Association, Including those at the Margate Congresses of 1935 and 1936, when Capablanca was one of the players and the boys of Chatham House School acted as the pieces. One of the smallest lads fainted at one of the displays, an occupational hazard of the unmoved Rook's Pawn. Most spectacular was the Hurlingham display of 1936, when games were played between Oxford and Cambridge, the House of Lords and the House of Commons, the Army and the Navy, the British Girl Champion and the British Boy Champion. The pieces marched on and off to musical accompaniment. The Kent Association also assisted in the Croydon display of 1939, when Lord Dunsany and Vera Menchik were among the players.

Other displays coincided with great events, such as the corporation in 1937. The Southern Counties' Chess Union organized a display for the Festival of Britain. Onlookers numbered 5,000. The boys of East Moor School, Leeds acted as the pieces and wore costumes of red and gold made by themselves. Players included Rossolino, Broadbent, Dr. C. E. M. Joad and Godfrey Nicholson M.P.

Further afield, one of the best contested and most spectagenic games of all time must have been the one at the Moscow Sport Palace in March 1962. This was between the reigning World Champion, Botvinnik and ex-World Champion, Smyslov. The pieces were ballet dancers. The result was a draw.

E. PRITCHARD

("Intelligent" Computer games welcomed by this department. Address all correspondence to COMPUTER GAMES DEPARTMENT, Personal Computing.)

Early GO Programs

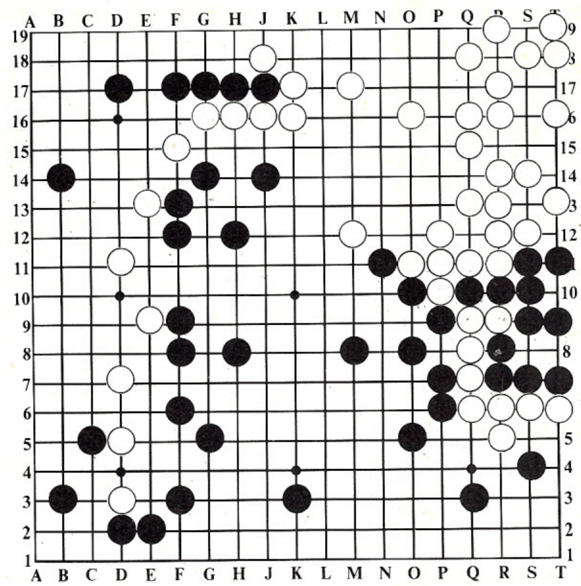
The earliest computer program to play a **complete** game of GO was written by Al Zobrist of Pasadena, CA., as a thesis for his PhD in 1968. Bruce Wilcox, co-author with Walter Reitman of the U. of Michigan's Reitman-Wilcox GO program, described Zobrist's initial efforts and also discussed GO programs in general in a recent "Computer Go" article. The material, copyright 1978 by Bruce Wilcox, appeared in the official Journal of the American Go Association, PO Box 397, O Chelsea Station, NY, NY 10011. (Wilcox's complete discussion, only part of which is reprinted here, deals with the problems involved in programming GO as opposed to chess.)

Only 5 years ago the world's foremost Go organization, the Nihon Kiin, published this statement:

"If it is possible for the tremendous power of the computer to conquer the game of Go, everybody will lose interest in it. Such apprehension, however, is completely groundless.... Fortunately, Go is such a profound and extensive game that it can never be controlled by machinery.... Those who are concerned in the computer business know it is only a fantastic dream to make the computer play Go in its own way in place of human beings."

In 1977, an introductory Go book stated that: "the game is said to be beyond the capacity of computers."

Well, I have been patient too long. I will assume these statements were made out of ignorance — but I've had enough of them! I am a skilled programmer and player; and I have no doubt that computers will play amateur shodan Go within my lifetime. (That gives me lots of time to program it myself. Twenty years is a more reasonable guess.) Nor should there be any concern over people losing interest if computers can play well. Until a computer is programmed with gamesmanship and showmanship — until a program recovers a losing game by deliberate swindle — there will always be a demand for Bobby Fischers. Far from hurting the game, computer programs seem to add new interest.



Position in a game of GO used by Zobrist to demonstrate his theory of influence numbers.

The sheer size of the Go board has a heavy impact on programming. Even at the end of a game of Go, there may be over a hundred legal moves remaining. Exhaustive searching in Go reaches computer time and space limits before going anywhere near the depth reached by chess programs. This is critical because to play good Go you must look DEEPER than in chess. Chess masters only occasionally examine lines of play going 15-20 moves deep. Go masters do this routinely, and often go much deeper. This does not mean Go masters are smarter than chess masters. Because of the smaller scale of chess (in pieces, game length, and space) and the mobile nature of the game, significant changes occur rapidly. The whole board can undergo upheaval in a few turns. Go, on the other hand, is played by successive addition of a single stone to a large board with the previous stones remaining untouched (except by capture). Thus the game exhibits fewer changes per move and longer sequences accomplishing even simple local tasks. The pace of Go is much slower and tactical interactions are often largely devoid of global impact.

Humans look 15 or more moves deep in chess because of a single-minded purpose which computer programs lack. The intent of computer lookahead is to find a move leading to an overall "better chess position", taking into account whatever factors it can. Looking at all sequences up to a depth of 6 or 7 moves finds many of the normal material and positionals gains that a human would find. (Of course the occasional misses of a 9 move sequence can prove fatal.) Such random searching finds interesting boards at a reasonable cost for chess, but cannot be transferred to Go. Because it takes many moves to bring about significant changes in a Go position, finding interesting results "by accident" is prohibitively expensive. Searches must be restricted to determining if and how a specific task can be accomplished. This is how humans control their searching in both chess and Go, enabling great depth to be achieved with few extraneous moves.

Chess and Go are both simplified example of "inexact problems", ones

that have no clear solution algorithm. Humans can be quite skilled at these problems, even if it is not clear how they do it. In programming any game, the temptation is to let the computer do its thing — grind out an answer by looking at as many moves as can be afforded. Doing this, chess programmers have achieved human-like performance without human-like thought processes.

Chess has long been a favorite domain for research in the field of Artificial Intelligence (AI), but Go may yet supercede it. As Berliner has noted:

"Human chess players largely delude themselves in believing that chess is a 'conceptual' game . . . . while chess may have to be 'conceptual' to a human processor, such an approach may not be at all necessary for super-fast machines. However, even if a full-width search program were to become World Chess Champion, such an approach cannot possibly work for Go, and this game may have to replace chess as the 'task par excellence' for AI. (For those desiring more on computer chess, I recommend Prof. Hans Berliner's review of chess program history and literature in ARTIFICIAL INTELLIGENCE, vol 10, 1978.)

Go is a whole new challenge requiring insights into human thought and new programming techniques. Because global changes occur slowly in Go, the game is much better suited to studying complex information management and decision-making than is chess. Aside from its serious merits, programming Go is fun. And since the field is still in its infancy, there are many opportunities to explore the unknown.

Having shown why programming is so hard, let us see just what has been accomplished.

November 17, 1968 — The earliest program to play a complete game of Go was written by Albert Zobrist for his PhD. (his thesis was entitled *Feature Extraction and Representation for pattern Recognition and the Game of Go*). There was no clear way to distinguish the human from the computer then. Both could be human. That game, between two complete novices, was not Go as we are used to it. It was, however, a landmark. The program (playing white) was able to play a complete game of Go (to move 232), and win by 7 points.

July 11, 1970 — The crucial test of a program's understanding of the game is seen in its play against a skilled player. This is the last version of Zobrist's program, playing Black with 13 stones against an estimated 7 kyu. Black's moves seemed almost human. (I say almost because, while they reflected human pattern knowledge, they did not reflect human understanding of the game.) The program did not "understand" why it played a move, so its moves failed to support each other.

Zobrist's program provided basic analysis of the board in terms of:

1) Segmentation of the board into Black and White regions. This was done using an "influence function". The idea is that stones control empty points near them. Black stones radiate positive influence; White stones radiate negative influence. Opposing influences cancel out, and whoever has more influence is left in "control" of that point. A segment consists of areas of contiguous points of the same sign (also called a group or army). The absolute value of the influence number at a point measured the degree of control over that point by a player's stones. A separate count was kept for the number of stones and the number of empty points contained in a segment.

An "influence function" is a common building block of most programs. It is often used to define groups and to handle potential territory and influence (walls, thickness, etc.). Mathematicians and programmers have had this fascination with influence functions, and I understand that Manfred Wimmer (former European champion and a professional Go player) worked on one around 1970.

Zobrist's influence function is worth a closer look. Consider a 19×19 matrix representing points of a Go board. Each Black stone was give +50, each White stone −50, and all other locations were given 0. Then each positive location added 1 to each of its 4 neighbors. The 1 point add/subtract process was repeated 3 more times. In effect this spreads influence of stones out to nearby areas, and the results are quite interesting. The accompanying diagrams, from Zobrist's thesis, shows a board position, its segmentation, and influence numbers generated (negative numbers are underlined).

2) Occupation of points and neighbors. Information was maintained as to whether a point was occupied, by whom, and how many of its neighbors were of each color (both adjacent and diagonal).

3) Stones and dame. (Dame = vacant intersection adjacent to a stone.) For each occupied point the program kept the number of stones in its chain and the number of dame in the chain. Each vacant point had associated with it

0	2	4	5	6	6	4	1	$\underline{7}$	$\underline{7}$	$\underline{6}$	$\underline{5}$	$\underline{5}$	$\underline{5}$	$\underline{7}$	$\underline{10}$	$\underline{59}$	$\underline{12}$	$\underline{57}$
2	4	8	10	10	11	11	2	$\underline{50}$	$\underline{12}$	$\underline{10}$	$\underline{10}$	$\underline{9}$	$\underline{9}$	$\underline{10}$	$\underline{62}$	$\underline{16}$	$\underline{63}$	$\underline{61}$
3	7	10	62	10	57	57	56	42	$\underline{56}$	$\underline{13}$	$\underline{62}$	$\underline{12}$	$\underline{11}$	$\underline{12}$	$\underline{14}$	$\underline{63}$	$\underline{14}$	$\underline{11}$
5	8	10	6	0	$\underline{4}$	$\underline{56}$	$\underline{57}$	$\underline{56}$	$\underline{64}$	$\underline{12}$	$\underline{12}$	$\underline{12}$	$\underline{62}$	$\underline{13}$	$\underline{64}$	$\underline{64}$	$\underline{14}$	$\underline{59}$
7	10	8	0	$\underline{7}$	$\underline{56}$	$\underline{7}$	$\underline{6}$	$\underline{6}$	$\underline{5}$	$\underline{8}$	$\underline{9}$	$\underline{9}$	$\underline{11}$	$\underline{12}$	$\underline{63}$	$\underline{15}$	$\underline{13}$	$\underline{10}$
8	62	6	$\underline{3}$	$\underline{6}$	1	56	8	57	3	$\underline{3}$	$\underline{6}$	$\underline{8}$	$\underline{8}$	$\underline{11}$	$\underline{14}$	$\underline{64}$	$\underline{63}$	$\underline{11}$
7	9	1	$\underline{7}$	$\underline{54}$	56	14	13	12	5	$\underline{4}$	$\underline{10}$	$\underline{8}$	$\underline{10}$	$\underline{12}$	$\underline{63}$	$\underline{65}$	$\underline{16}$	$\underline{59}$
2	0	$\underline{3}$	$\underline{11}$	$\underline{6}$	58	13	62	10	2	$\underline{7}$	$\underline{58}$	$\underline{5}$	$\underline{12}$	$\underline{63}$	$\underline{16}$	$\underline{65}$	$\underline{56}$	$\underline{4}$
$\underline{1}$	$\underline{4}$	$\underline{10}$	$\underline{62}$	$\underline{6}$	6	11	10	7	1	$\underline{2}$	0	47	$\underline{49}$	$\underline{66}$	$\underline{57}$	$\underline{50}$	50	54
$\underline{2}$	$\underline{5}$	$\underline{9}$	$\underline{12}$	$\underline{7}$	6	10	9	6	3	2	7	12	48	$\underline{42}$	42	50	65	12
$\underline{1}$	$\underline{4}$	$\underline{8}$	$\underline{12}$	$\underline{54}$	56	12	11	8	6	8	10	12	14	48	$\underline{50}$	$\underline{42}$	57	60
$\underline{2}$	$\underline{5}$	$\underline{9}$	$\underline{11}$	$\underline{5}$	58	13	62	10	8	10	62	12	62	8	$\underline{51}$	49	15	11
$\underline{1}$	$\underline{3}$	$\underline{7}$	$\underline{61}$	$\underline{4}$	8	12	10	8	7	8	10	11	13	56	$\underline{50}$	50	57	53
3	3	0	$\underline{8}$	$\underline{3}$	58	12	10	7	5	6	8	10	13	56	$\underline{57}$	$\underline{58}$	$\underline{57}$	$\underline{53}$
6	11	53	$\underline{54}$	$\underline{1}$	9	62	10	8	7	7	7	10	62	7	$\underline{2}$	$\underline{58}$	$\underline{7}$	$\underline{4}$
8	12	6	$\underline{4}$	1	11	12	10	10	10	8	8	8	10	12	5	6	55	3
8	61	6	$\underline{44}$	5	62	11	9	10	62	10	6	6	8	10	62	11	11	7
7	11	11	56	63	12	8	6	8	10	8	4	3	4	8	10	9	7	4
4	6	8	9	9	7	4	3	4	5	4	2	0	2	4	5	5	3	2

The Zobrist "game position" showing derived influence numbers. Black stones are valued at +50; White stones are −50. Values neighboring stones are determined by their locations.

the number of empty adjacent intersections.

With that board information, here is how the program worked. There was a set of "patterns" to be applied to the entire board in all areas and orientations. These patterns were lists of tests for the existence of certain values in the various basic board data at specified locations. One such pattern might be the equivalent of: Black stone at (0,0) with 1 dame, empty space at (1,0), and Black stone at (1,1) with at least 4 dame = the play at (1,0) is worth 500 points. This would detect chains in atari (equivalent to check in chess) which could be saved by connecting them to another chain. The program looks at each intersection of the board, one by one. If a Black stone with 1 dame is found (the 0,0 relative coordinate base), then the program looks for a neighboring empty point (location 1,0) and corresponding safe Black string (location 1,1). When instances of this pattern are found, a 500 point bonus is added to the proposed move location, the connection at (1,0). This is an example of specialized knowledge that would not be matched very often. Consider the pattern: White segment at (0,0), Black segment at (1,0) = the play at (0,0) is worth 40 points. It is a general one that might match 100 times in a single pass over the entire board and gives the program an "urge" to advance toward the enemy.

Many patterns were applied to the board, with the scores being summed for each location. Whichever ended up with the highest score became the program's next move. (Notice this is like "whole-board" evaluation after a search of all moves to a depth of one.) Some patterns were used only in certain "phases" of the game, to deal with differences, say, between fuseki and endgame. Other patterns were used to downgrade moves. Ko (an infinite recapture situation) was a recognized pattern and a bonus of -4000 was given an illegal ko recapture. This effectively removed the move from consideration.

Patterns alone didn't play a reasonable game of Go (not surprising), so lookahead was added. The program used some of its patterns to mark points where lookahead should be performed. The lookahead was extremely weak, and was used for connectivity and stone safety. Brute force was the key—look at all combinations of moves within the range of marked points; but only to a depth of 3 moves. There was a separate routine to handle ladder searches, which went to an indefinite depth (the only allowed moves were atari and save). Moves which accomplished the lookahead goal had bonuses added to them, just as though they had matched a pattern (in a sense they had). To handle life and death situations (tsume-go), rather than use normal lookahead, a special mechanism attempted to determine sets of points needed to form two eyes if unopposed. Moves entering into the greatest number of eye formations were given bonuses if there were neither too few nor too many eyes formable.

Clearly his program was a weak Go player. But if you added a good lookahead system and more patterns, how far could it develop? I don't know. In some sense you can characterize all Go play as pattern and lookahead controlled. But his program design had several important omissions. Humans use patterns that make sense only with the results of earlier computation (patterns of patterns of . . .). "Push your opponent's weak stones toward your strong ones" is a pattern statement which first requires using lookahead and patterns to recognize weak and strong stones.

For the program's moves to look reasonable so frequently, clearly there must be something to the notion that humans use patterns to guide their play. But while the patterns had motivations implicit in them, no testing was done by the program to see if the implied motive was reasonable in the current situation. Nor was there any continuity of purpose from turn to turn. Humans filter out many otherwise reasonable pattern moves because one pattern is more valuable for a specific on-going goal. One thing all progams that I know of have in common is an inability to play out a ko fight. They recognize that it is illegal to recapture immediately, but since ko is a motivational concept, they know nothing about generating ko threats. Instead the program merely selects its next best move. (Many ko threats are stupid moves in and of themselves.)

While Zobrist's program left much undone, remember that his was the first. It is thus an important milestone in computer Go. And it beat human beginners, setting the stage for improved programs.

(Part II of the Zobrist GO program will show one of the games played by this program together with analysis by Bruce Wilcox. Also included will be some personal observations received from Al Zobrist in some recent correspondence.)

A recent note from Bruce Wilcox said he was leaving University of Michigan's GO project. (The program will be continuing at the University.) His future plans indicate that he will become involved in some interesting developments in the growing field of Computer Go. "Depending on where I end up" he writes, "I will either be doing a conversion of our program for a big machine, (aiming to make it more transportable to any machine,) or I will be intensively exploring microcomputers and will pursue the possibilities of producing a micro GO-program." Whichever path Bruce follows, he is sure to provide some valuable and long-awaited contributions to computerized GO. His new address is now "Inometrics", 700 Concord Ave., Cambridge, MA.

Background on GO

Professor David W. Erbach, of the mathematics department, University of Wisconsin at Madison, describes himself as "neither the best computer expert nor the best GO player in the country. But I am probably as familiar as anyone on both." He offered an interesting and authoritative look at GO in a recent letter. Extracts from that letter follow:

"The game and its history: As you may know, the origins of the game are lost in antiquity. It was originally Chinese, and in old days, along with calligraphy and flower arranging was considered one of the three classic arts.

It came to Japan, along with the characters and with Buddhism, around the 7th century, and has been played there steadily ever since. With minor modifications (e.g. the board size was increased from 17 × 17 to the present 19 × 19) the rules have remained the same, and as such, it must surely be one of the oldest essentially unchanged human intellectual activities. There exist fragmentary game records which are very old, and essentially complete game scores from the 12th century. I personally find it quite remarkable that one can go back and reconstruct the exact thought processes of someone living in such a remote epoch!

"The Japanese are now the most avid players, and support several hundred professionals. There the situation is rather comparable to say tennis or golf in this country, with a hundred or so people who can live from tournament winnings, and others who survive from teaching. The most important tournament, the Kisei, has a total prize fund of around $1 million, and is held annually. The major professional association is the Nihon Kiin, in Tokyo, which has a large building (including hotel rooms), and which certifies players and controls grades. There is also the somewhat smaller Kansai Kiin, which has headquarters in Osaka.

"The rating scale is in Kyus and Dans. An absolute beginner starts around 30 Kyu, and grows in strength up to 1 kyu. The weakest Dan grade is 1 dan, and they progress in strength up to 6 or 7. These are amateur grades, which are used world-wide, and an attempt is made to keep them in international alignment. In addition, there are the professional grades, increasing in strength from 1 to 9 dan. Professional 1 dan (or shodan as it's usually called) is a little stronger than the strongest amateur grade, and they move up very substantially. Even full-time pros cannot hope to advance much faster than a grade every other year, and most never will reach 9 dan.

"The Chinese and Koreans both have some players of professional strength, but westerners don't look too well. After many years in the professional training school, last year an Austrian, Manfred Wimmer, and an American, James Kerwin, were given ratings of provisional professional shodan. Wimmer was soon promoted to 2-dan, and Kerwin has since won the 1-dan division of the Kisei Elimination Tournament in Japan. Thereby, he achieved the first win by a Westerner in a professional Go tournament.

"Go and computers: For some years, there have been attempts to get computers to solve tactical Go problems, for instance about life or death of small corner groups. The whole-game program in this country was part of a Ph.D. thesis by Albert Zobrist ("Feature Extraction and Representation for Pattern Recognition and the Game of Go.") The next was by Jon Ryder, in another Ph.D. thesis at Stanford, ("Heuristic Analysis of Large Trees as Generated in the Game of Go.") 1971. Unfortunately neither of these is able to defeat anything but the rankest beginner, and as long as the analysis is on a 19 × 19 board, it is not clear how soon that will change. Each of them showed the influence of chess and checker analysis, namely the attempt to extract certain salient features of a position, take a weighted sum, and choose the move with the highest score. A more elaborate description of Ryder's work has since appeared in the American Go Journal.

"The strongest player anywhere interested in computing, as far as I know, is Jon Diamond, a 6-dan player, who recently retired as undefeated British champion. He is still active professionally in computing. In addition, in this country, David Benson, Prof. of Computer Science a Washington State U., Pullman, WA 91163, has a number of graduate students working on the problem.

"For various reasons, Go seems to me to be a better game than chess for AI work. For one thing, the rules are less artificial. After all, you could change the initial starting position, or exact move descriptions in chess, and have a game about as good. All that must be programmed around, but one isn't learning from it. Also, the opening is not nearly so stereo-typed, and there is nothing like the business of reading books for 10 moves. Finally, because not everything rides on a single event like mate, but the accumulation of territory, the best move may float from move to move around the board, and one must constantly watch the growing together of a dozen or more positions. Go, is simply an astronomically larger game than chess (say an average of 200 legal moves going on for 200 half-moves, in contrast to say 100 moves over 100 half-moves.)

"The strongest current programs can scarcely beat someone who has just learned the rules. But there is one thing that might help, and that is to reduce the board size. If one plays fairly well, one eventually becomes persuaded that 19 × 19 is the best size board. But when one is learning, it's much better to start say on 9 × 9. Otherwise the period between incorrect move and punishment may be 100 moves, and that makes it hard to learn. But, while less interesting for an expert, the game is largely the same on the smaller board, and that might make enough reduction in exhaustive searches to make a reasonably strong program feasible. I suspect that with substantial but reasonable effort, it might be possible to put such a thing on a few chips, or a micro, and there might well be some money to be made by someone with the determination to do it. I hope some of this will be found interesting or useful by your readers."

A New Micro Program for Go on Cassette

Hal Muller of Toronto, Canada, who teaches computer science and mathematics, has written what is apparently the first GO program for a microcomputer. The routine, which runs on an 8K PET, is a modified version of regulation GO. its main modification is the use of a 9 × 9 grid instead of the 19 × 19. "But otherwise," says Professor Muller, "the program follows all the rules and logic of GO." (See his ad in the classified section of the Computer Chess Department.)

Old Hands and New Deals

BY THOMAS A. THROOP

This column is being devoted to a discussion of some deals as bid and played by the Bridge Challenger of Fidelity Electronics. The play of the Challenger will be compared to the play of my own bridge playing program. Also, I am passing along some comments received from Jim Hilger on the performance of his program with some deals discussed in this column.

For those of you who are new readers, the general characteristics of Fidelity's Bridge Challenger were discussed last year in the July, September and December issues of PERSONAL COMPUTING. As mentioned in those columns, Bridge Challenger will bid and play one, two, three or even all four hands while human players bid and play the remaining hands.

The following deal is from the 1979 November bridge column:

COMPUTER NORTH (Dummy)
♠ QJ10532
♥ KQ
♦ AQ974
♣ —

WEST
♠ K9864
♥ J8
♦ J6
♣ A543

EAST
♠ A7
♥ 653
♦ K1032
♣ 10982

COMPUTER SOUTH (Declarer)
♠ —
♥ A109742
♦ 85
♣ KQJ76

With South as dealer, and Bridge Challenger playing the N-S cards, the bidding by the computer device is as follows:

Bidding

South	West	North	East
1H	Pass	2S	Pass
3C	Pass	4S	Pass
5H	Pass	6H	Pass
Pass	Pass		

Looking at the North-South cards, the slam contract is a bit ambitious. To make the slam with good play requires that West hold the diamond king, that the heart jack be a doubleton, that the club suit be divided 4-4 or certain other favorable club holdings exist with correct guessing on your part, and that a trump not be opened by West when East holds the ace of clubs.

The play of the cards, with my playing the E-W cards, is shown in the accompanying tableau. I opened the jack of diamonds from the West hand. Somewhat to my surprise, the Challenger did not recognize the finesse combination in diamonds and incorrectly went up with dummy's ace. It then played two rounds of trump, not realizing that one club must be ruffed in dummy at some point. The Challenger then entered declarer's hand with a spade ruff and drew the last enemy trump. At trick 6 Challenger attacked the club suit, leading the king to drive out the ace. I won with West's ace and returned a spade. The computer played the 5 from dummy, I necessarily played East's ace, and Challenger ruffed. The electronic unit continued playing clubs and lost a trick to East's 10 at trick 10. I then cashed East's king of diamonds. The last two tricks were won by N-S, for a final result of 10 tricks won by N-S and 3 won by E-W.

	W	N	E	S
Trick 1	JD	AD?	3D	5D
2	8H	QH?	3H	2H
3	JH	KH?	5H	4H
4	4S	2S	7S	7H
5	6S	3S	6H	AH
6	AC	4D	2C	KC
7	8S	5S	AS	9H
8	3C	10S	8C	QC
9	4C	7D	9C	JC
10	5C	JS	10C	7C
11	6D	9D	KD	8D
12	9S	QD	10D	6C
13	KS	QS	2D	10H

Tricks N-S (Challenger): 10
Tricks E-W: 3

The play of this same deal by my program was discussed in detail in the 1979 November column. On two significant points its play differed from that of the Bridge Challenger. First, at trick 1, my program recognized the finesse situation in diamonds and finessed North's queen. Second, my program recognized the need for a club ruff at some point and took a ruffing club finesse with king of clubs at trick 4. The finesse was continued at trick 5 when West ducked with the 3 of clubs at trick 4.

The complete play of the deal by my program is shown below. Eleven tricks were won by the computer as N-S; E-W won only the diamond king and one additional trick with the 10 of clubs.

	W	N	E	S
Trick 1	JD	QD	KD	5D
2	8H	KH	3H	2H
3	4S	2S	7S	4H
4	3C	3S	2C	KC!
5	AC	QH	8C	QC
6	6S	5S	AS	7H
7	JH	4D	5H	AH
8	8S	7D	6H	10H
9	4C	10S	9C	JC
10	5C	9D	10C	7C
11	6D	AD	2D	8D
12	9S	JS	3D	10H
13	KS	QS	10D	6C

Tricks N-S (computer):11
Tricks E-W: 2

The following deal was generated by my dealing routine on the day I prepared this article. The deal which was given to both Fidelity's Challenger and to my program was as follows:

COMPUTER NORTH (Dummy)
♠ 10632
♥ A982
♦ K3
♣ K109

WEST
♠ J84
♥ 7
♦ A106
♣ AJ8753

EAST
♠ Q75
♥ KJ10653
♦ 842
♣ 6

COMPUTER SOUTH (Declarer)
♠ AK9
♥ Q4
♦ QJ975
♣ Q42

South was assumed to be the dealer with Bridge Challenger playing the N-S

cards. The bidding by Challenger, with no bidding by the E-W hands, was as follows:

Bidding

South	West	North	East
1D	Pass	1H	Pass
2NT	Pass	2NT	Pass
3NT	Pass	Pass	Pass

As West, I made the normal opening lead of the 7 of clubs against the computer's 3-no-trump contract. Challenger played the 9 from dummy, which won the trick. The unit now made a serious blunder, dooming the contract to defeat; the spade suit, rather than the diamond suit, was attacked. The 10 of spades was led from dummy (if spades are led, the deuce, rather than the 10, should be led). I followed with East's 5, Challenger played South's 9, and I won with West's jack. I now played West's ace of clubs and followed with the 3 of clubs, clearing the last stopper from the N-S hands. The Challenger now turned to the diamond suit, but it was too late. When I won North's king with West's ace, I was able to cash three club tricks thereby setting the contract by two tricks. On the jack and 8 of clubs the Challenger made incorrect discards, throwing possible winning diamonds rather than a losing heart. On the 5 of clubs the Challenger finally discarded the losing heart. Tricks 9-13 were won by Challenger's cashing its remaining winners. The complete play of the deal is shown in the accompanying tableau.

		W	N	E	S
Trick	1	7C	9C	6C	2C
	2	JS	10S?	5S	9S
	3	AC	10C	6H	4C
	4	3C	KC	3H	QC
	5	AD	KD	2D	5D
	6	JC	2S	4D	7D?
	7	8C	3S	8D	9D?
	8	5C	6S	5H	4H
	9	7H	AH	10H	QH
	10	6D	3D	JH	QD
	11	10D	2H	KH	JD
	12	4S	8H	7S	AS
	13	8S	9H	QS	KS

Tricks N-S (Challenger): 7
Tricks E-W: 6

After finishing this deal with Fidelity's Challenger, I gave the same distribution with the same final contract to my program for play. As West, I again opened the 7 of clubs in order to present my computer declarer with the same situation as faced by the Fidelity declarer. My program also played the 9 from dummy, winning the trick in the same manner as the Challenger. However, now at trick 2, my program properly attacked the diamond suit with the lead of the king of diamonds from North. I won with West's ace of diamonds and then played the ace and another club to drive out the last N-S stopper. My program now ran the diamond suit, won the ace of hearts and the ace-king of spades to make 3 no-trump, whereas Challenger went down 2 tricks. The complete play is shown in the tableau below:

		W	N	E	S
Trick	1	7C	9C	6C	2C
	2	AD	KD	2D	5D
	3	AC	10C	6H	4C
	4	3C	KC	3H	QC
	5	6D	3D	4D	QD
	6	10D	2H	8D	JD
	7	7H	2S	5H	9D
	8	5C	3S	10H	7D
	9	8C	AH	JH	4H
	10	4S	6S	5S	AS
	11	8S	10S	7S	KS
	12	JS	8H	QS	9S
	13	JC	9H	KH	QH

Tricks N-S (Throop program): 9
Tricks E-W: 4

In December 1979, at the Greater New York Bridge Association Regional Tournament, I was scheduled to demonstrate my program together with Fidelity's Challenger, in the play of several deals which each program will see for the first time. The results of this match will appear in my March column.

A new bridge bidding and playing program has been written by Jim Hilger of 5315 17th Ave., Moliner, IL 61265. Jim, who owns an Apple II computer with a floppy disk, has written two major bridge programs in "integer basic". These programs, a BID program and a DEFENSIVE PLAY program, each require about 16K of memory. These two programs, together with some very small programs to link the two major programs, form a bridge system he has named "TRIK 1.0".

The user of TRIK 1.0 is first asked whether he would like to play a deal previously saved on the disk or play a new deal. The bidding program then bids for North, East, and West, while the user bids the South hand. The dealer is determined randomly by the program. At the conclusion of the bidding, if the final contract is a N-S contract, then you may play the N-S cards while the defensive play program defends against the contract, just as does the Duisman program.

Jim has sent along several examples of deals bid and played by his TRIK 1.0. Two of them are deals which I discussed in my 1979 August column. These are deal #32 of set 3.65 as generated by the PET version of the Duisman program and the example of Wasserman's bidding program. The bidding and defensive play of deal #32 and the bidding of the Wasserman deal by Jim's program are shown in the accompanying tableaus.

NORTH
♠ J76
♥ AK85
♦ Q104
♣ 1032

WEST
♠ 8532
♥ J1063
♦ 52
♣ 874

EAST
♠ KQ
♥ 9742
♦ K8763
♣ KJ

SOUTH
♠ A1094
♥ Q
♦ AJ9
♣ AQ965

BIDDING (West dealt.)

(YOU) SOUTH	WEST	NORTH	EAST
	Pass	Pass	1D
Double	Pass	1H	Pass
3C	Pass	3N	Pass
Pass	Pass		

PLAY:

	W	N	E	S
Trick 1	S2	S6	SK	SA
2	S3	S7	SQ	S10
3	D2	D10	D3	D9
4	C4	C2	CJ	CQ
5	H3	H5	H2	HQ
6	C7	C3	CK	CA
7	S5	SJ	H4	S4
8	D5	DQ	D6	DJ
9	H6	HA	H7	C5
10	H10	HK	H9	C6
11	C8	C10	D7	C9
12	S8	D4	D8	DA
13	HJ	H8	DK	S9

Made, 3 overtricks.

Sample hand from Wasserman's "Bid Program" (described in a previous issue.)

NORTH
♠ AQ64
♥ 1096
♦ 53
♣ Q765

WEST
♠ J97
♥ AQ852
♦ 1062
♣ A8

EAST
♠ 3
♥ 3
♦ AKQJ874
♣ J1093

SOUTH
♠ K10852
♥ KJ74
♦ 9
♣ K42

BIDDING (West dealt.)

(YOU) SOUTH	WEST	NORTH	EAST
	Pass	Pass	1D
Pass	1H	Pass	2D
Pass	2N	Pass	3D
Pass	4D	Pass	Pass
Pass			

PLAY: N/A

Regarding the Duisman deal #32, TRIK 1.0 (and Jim) reached my original proposed contract of 3 no-trump. As West, TRIK 1.0 opened the 2 of spades against the 3 no-trump contract, Jim played the 6 from dummy, the program played East's king, and Jim won with the ace. In order to reach dummy with the jack of spades, Jim now led the spade 10 to drive out the queen. As East, the program then led the 3 of diamonds at trick 3, after winning the queen of spades. Jim played the 9 from South and won with North's 10 to reach the dummy for the club finesse. The club suit was very friendly and Jim eventually won twelve tricks.

In the case of the Wasserman deal, Jim passed with the South hand after TRIK 1.0, playing East, opened 1 diamond. The program reached 4 diamonds with the East-West cards. Jim bid the deal a second time with his program, this time doubling 1 diamond with the South hand to be consistent with the Wasserman sequence, although South does not have the values for a takeout double. TRIK 1.0 then redoubled as West, bid 1 spade for North, and 2 diamonds for East, just as did the Wasserman bidding program. Jim then bid 2 spades for South to follow the Wasserman sequence, after which TRIK 1.0 bid 3 hearts for West, pass for North, and 4 diamonds for East. With 27 points in the combined E-W hands and the solid trump suit, 4 diamonds is quite a conservative contract. Five diamonds, which was reached by the Wasserman bidding program, can be made against any defense.

Next month I plan to discuss more of the deals sent in by Jim. For those of you with questions about any of the deals appearing in this column, please let me hear from you. Also, remember to send in your responses to the bridge survey that appeared in the October issue. — *T.T.*

Computer Power for Small Businesses

Computer Power for the Small Business, by Charles J. Sippl and Fred Dahl; Prentice-Hall, Inc., Englewood Cliffs, NJ 07632; 306 pp.; $6.95 paperback.

Undoubtedly, small computers can add to the operating efficiency of small businesses. But matching a computer system to specific business needs can be difficult if a person doesn't understand the fundamentals of buying a computer. According to Sippl and Dahl, "If today's small businesspersons and middle managers can develop the familiarity and willingness, then the computer can ... be transformed from a risky weapon to an effective business tool."

Computer Power was written to familiarize business people with computers and help them develop a purchasing plan. The book is for people who must make the purchasing decision and ask: "Can my business benefit from a computer?"

Buying a computer, or knowing whether a business can benefit from a computer, requires some knowledge of how they work. The book begins with a condensed yet easily digestible description of computers and their operation. According to the authors, business people only need to understand the basic terminology and fundamental criteria for buying computers.

Next, the authors describe a number of computer systems available for business applications. These systems are arranged in order of size and cost.

In the chapter on software, readers can find ways of obtaining programs for their needs. Typical programs include inventory, payroll, general ledger, balance sheet and mailing lists. A person can learn the computer language and write programs; buy general, "canned" programs from software producers; or hire consultants to write custom programs for specific needs.

The authors included three helpful tables in these beginning chapters. One table lists 74 computer systems, with information such as memory capabilities, type of central processing unit, basic system cost and additional features. Another table lists the types of programs — and their applications — offered by computer system manufacturers. The third table lists 22 other program sources, with address and product descriptions. While these tables may not be complete, they make a useful guide to knowing what's available.

After orienting the reader to computer systems and applicatins, Sippl and Dahl address the problem of deciding whether to buy the computer. (The computer's cost-effectiveness may not justify the system's cost to the business.) They begin by outlining a plan to make the purchasing decision and then how to choose a particular system. The plan basically covers three points:

- Define the need by looking at problem areas in the business "where computerization may be a source of improvement"
- Evaluate whether the available software/hardware can fill the need
- Select the system based on criteria previously outlined in the book

Also included are the implications of owning a computer, such as the effect on personnel.

The final chapters discuss cases of computers in businesses ranging from law firms to libraries, and alternate uses of computers in automation, energy conservation and security.

In their book, Sippl (a former instructor and lecturer in computer electronics) and Dahl (a small businessman) use a light and conversational writing style. They also write in terms of the business — applications, cost and efficiency. And with their background and an ample supply of illustrations, the authors provide the specific information and insight that a decision maker needs. —*Joe Bobbey*

An Industrial Revolution

Running Wild — The Next Industrial Revolution, by Adam Osborne; Osborne/McGraw-Hill, Inc., 630 Bancroft Way, Berkeley, CA 94710, 181 pages, $3.95 paperback.

- Radio Shack sold one hundred million dollars worth of microcomputer systems in the first 18 months the TRS-80 was available.
- Apple Computer, founded by 24-year-old Steve Jobs and 20-year-old Steve Wosniak, who designed their microcomputer for the fun of it, will probably gross more than fifty million dollars in 1979.
- Alpha Micro Systems, founded in 1977 by three men with $50,000 worth of equipment and $1000 in cash, sold approximately five million dollars worth of computer systems in 1978 and may have doubled that in 1979.
- Because of new microelectronic innovations, the post office as we know it today will disappear within the next 20 years — possibly within the next ten.
- Soon IBM will be facing stiff competition, not from its traditional rivals such as Burroughs, Univac, Honeywell and NCR, but from such upstarts as Apple Computer Corporation, Radio Shack, Commodore, Pertec, Texas Instruments, and the whole wolf pack of little guys.

Sound incredible? It is. It's only the beginning.

Dr. Adam Osborne, internationally known author, lecturer, spokesman and publisher in the world of microelectronics, a man who has been an integral part of the field during its rapid development over the past five years, scrutinizes the industry and projects its effects into the future. What he sees in his crystal chip will shake society as we've come to know it. Buck Rogers is alive and well in the 1980s.

Running Wild spells out how microelectronics got this far and gives knowledgeable insight into what's likely to occur next. But Dr. Osborne is no wild-eyed augur enchanted with the concept of microelectronics because with his predictions (stockbrokers, airline reservationists, communication workers and secretaries could be eliminated; medicine and education will be profoundly changed) come warnings of new horizons for thieves and embezzlers.

Running Wild will be of value to two segments of society: that portion which has, at some time, had contact with computers; and the more general

population who can't have helped but notice the proliferation of electronic games, computers for cars, electronic calculators, watches and sewing machines, general computer ads on television, the TVs themselves, microwave ovens and a myriad of other applications.

Huge volumes will be written some day explaining in retrospect the history, development and social impact of microelectronics; many things will be obvious by that time. Osborne's book puts you into the next industrial revolution as it's happening. Written in highly entertaining, anecdotal style, *Running Wild* explains technical concepts in layman's terms.

Next to your user's manual, *Running Wild* could tell you more about the power and future of your microcomputer than anything published to date. With its low price and readable text, this little book should be required reading from grammar school on up.

The revolution is upon us. Incredible change takes place every day, yet it's only a bit in the bytes to come. Osborne tells you where we are and forces you to think about where we're going. Your answers today will map the structure of tommorrow. —*Ken Mazur*

Getting Down to Business

Getting Down to Business with Your Microcomputer, by James A. Gupton Jr.; Sourcebooks, Northridge, CA 91324; 245 pp.; $9.95 paperback.

As hobbyists, homemakers and business people with little knowledge of computers find microcomputers entering their worlds, they need to know more than what computers are and how they work. *Getting Down to Business with Your Microcomputer* fills that need by specifically showing the usefulness of computers, what kinds are available and how to buy them. In his book, Gupton attempts to provide "a guide to the intelligent selection of computer components and aid in the assembly of your own personal computer system."

Gupton briefly introduces readers to microcomputers in the opening chapters. He describes the computing process and fundamental computer components. These descriptions include specific models (with illustrations throughout the book) and brand names. For example, in explaining how microprocessors work, he compares the major products, such as the Intel 8080A, Motorola 6800 and Zilog Z-80.

Next, the author compares selected entire microcomputer systems. Each system — those by Radio Shack, Heathkit, Altair, Apple, Ohio Scientific and five other producers — is described along with its special features. However, because Gupton does not discuss specific disadvantages of the systems, these chapters tend to sound like a collection of sales promotions. Also, the reader must rely on other sources to find out about available systems not included in the book. A complete list of addresses of the named system companies appears in an appendix.

Finally, the book offers suggestions on how to select a microcomputer for the home or business.

Other purchasing tips on what features to look for can be found in the product descriptions. But the author reminds: "The first step in choosing a microcomputer system is to determine for what it will be used and what level of performance these applications will require."

A highlight of this book is its resemblance to a general owner's manual. Gupton gives an idea of the care, operation and maintenance involved in owning a microcomputer. For example, in a discussion of audio tape cassette data storage, Gupton warns: "Do not rewind magnetic tape prior to storage. The fast motion will create stresses in the tape that may in time cause it to stretch. Instead, rewind the tape only before use."

Despite this specific and detailed introduction to microcomputers, beginners should feel comfortable reading it. The book is well written by the author of over 200 articles on electronics and photography. Also, technical terms are carefully defined. So as Gupton promised in his introduction, beginners should have a workable understanding of microcomputers after reading the book. —*Joe Bobbey*

PRODUCT CLOSE-UP

Top-Notch Software from The Bottom Shelf

BY KEN MAZUR

If you're looking for good, solid, professional software for your Level II TRS-80 at a reasonable price, check into The Bottom Shelf (TBS). I think you'll be glad you did.

In running Checkbook II, Info System, Basic Toolkit and System Doctor, I was impressed with the quality of programming. The displays are nicely formatted; the programs are fast. Best of all, the products, both applications and utilities, are useful.

The first program run was Checkbook II because I expected to be able to dispense with it rather quickly. Past checkbook programs I've used seemed to concentrate on cutesy graphics to represent checks. Some were more trouble than balancing a checkbook by hand. The TBS program was a welcomed change.

Requiring 16K or more, this program does everything necessary to keep your checkbook balanced and then some. No cutesy graphics here, just a simple, professional screen format that's easy to understand.

You initiate the program by typing 1 from the menu to input checks from the keyboard. Each of the columns of information (check number, date, to whom, amount and code) have little blocks lighted to show you the amount of information required. A check number can be up to five digits. Dates are automatically formatted into the form MM/DD/YY. If you enter 52871, the screen displays the entry as 5/28/71. There are 16 character spaces for the payee to whom you can make checks out up to $1,000,000. The last entry for each check is a four-character code of your choice. The codes come in handy when you use the Search and Total routine.

When all the checks are entered, type 99999 as the next check number and the computer automatically sorts the checks by number and returns to the menu.

Typing 2 from the menu brings the List and Edit function to the display. Eleven checks at a time are shown and an Enter shows the next eleven. Each of the five pieces of information per check may be altered with the Edit function. Tapping the "D" key followed by a check number removes the check from the file.

The real power of this program, however, comes not from check entry and display, but in the options of Print with Balance, Search and Total, Reconcile, Sort, and Check File Length. You also have available Input from Disk, Output to Tape, Output to Disk, Clear and Kill.

To use the Print with Balance option, you must enter the balance brought forward. After the first time, however, you will not need to enter the data again because the balance is protected and recorded in your data files.

Search and Total, my favorite option, enables you to pull out a check or series of checks that have any field (except amount) in common. The function lists those checks and gives a total. It's an effective way to see just what you're spending on any particular category.

Checking the status of your file is simple with the File Length option. After typing 9 from the menu, you are shown how many checks and deposits are in memory and how many more you can enter. The maximum number of checks for a 16K machine is 75; for 32K, 350; and for 32K with DOS, 150.

Reconcile is the last operation you should make during a session. After following the prompts, one of two things will happen. You will either get an error message that tells you the amount of the error or you will get a message saying, "Checkbook Balances."

There's more to this program but experimenting on your own will be fun and productive.

TBS's Information System, also designed for a machine with at least 16K, is basically an operator programmable, in-memory data base manager. Up to 10 fields are allowed with up to 40 characters per field to a total of 200 characters per record. The number of records held in memory depends on the number of fields you create, the field lengths and the amount of memory your computer has.

Once you have your data base established you can sort it by any field with a high-speed machine language sort. When the sort is complete, the program returns to the menu. Typing 5 (Video Display) lists the records sorted by the selected field. The sort routine is fast—and I do mean *fast*.

The complete menu (just to show you how many options you have) is: Add, Edit, Sort, Search, Video Display, Print, Read Data File, Write Data File, Initialize New Data File and End. All of the functions are easy to use with prompts from the program.

The high quality of the programming in general shows up throughout the program. For instance, hitting the Add function from the menu shows you what each record looks like, the total number of records your system will hold and the number of active records you have. Flashing blocks indicate where the information will be going in a record. The lengths of the fields are displayed also. You can not only sort, but can perform sorts within sorts.

For those of you with printers, there are two printing modes in the program. The first, a "screen print," automatically moves information from the display to the printer. The second is a more formal module accessed from the menu. It allows you to create printouts of the information on file in a format you desire. You can create mailing labels, Rolodex cards, index cards or just about anything else you wish.

With Information System, rewriting past filing programs or altering them to contain new fields is a thing of the past. When I want to file something now, I set up a new file using Information System and I have whatever fields I desire.

An added benefit, one not directly the result of the program itself, is the information contained in the program documentation. The documentation has one of the most lucid, most easily understood explanation of fields, records and files I've run into.

The next program I looked at was Basic Toolkit. It's a programmer's delight. For me, the Variables Map alone is worth the cost of the program.

I can still recall the day I added a subroutine to a previously written program. The original program was so long I had deleted the REM statements explaining the variables so I could get it into 16K. The program appeared to work fine in its modified form until the first time I called the subroutine. I couldn't believe the numbers on the display. I knew a bug had to be in the new subroutine. The problem was traced to a variable I had already used. That was the easy part. To avoid the same problem in the future, I had to list the program and hand-write each variable name as I came to it.

Those were the bad old days. I put that same program into my machine and used the Basic Toolkit on it. The display filled with an alphabetical listing of each variable used, a list of the lines the variable appears on and the number of times it appears on the line.

Merge is another option I was particularly pleased with. The function allows you to load multiple programs into your machine, concatenate programs and/or subroutines or execute one program while protecting an earlier one. If you have a favorite subroutine for sorting, you can keep it on a separate tape and add it to another program you've written; all you do is merge the two.

Other menu options:

•GOTO XREF examines the current BASIC program and lists in numerical sequence the destination of each GOTO and GOSUB statement and the line number it appears on

•RECALL allows you to recall a BASIC program after you hit Reset, erroneously type NEW or go to the DOS

•SEARCH MEMORY, an aid to assembly language programmers, looks for every occurrence of a two-byte combination and lists the location where it occurs

•TEST MEMORY checks your computer memory to determine if every location is operable

Basic Toolkit resides in memory while you are programming and can be accessed by hitting Shift/Break. A valuable aid to anyone who writes programs in BASIC, the Toolkit works with both disk- and cassette-based machines of 16, 32 or 48K.

I saved Systems Doctor for last because I have always felt uncomfortable with the hardware aspect of my computer. I could understand the theory of how and why it worked, but all those little square things inside the keyboard and the myriad of silver tracings intimidated me. If I had problems with my machine, I was at the mercy of the repair centers.

No more! Systems Doctor does a diagnostic check on the components of your computer system. SD lets you know if something is wrong before you program or enter data. With this program, you have some idea where a problem lies before going to the repair shop.

The main menu offers the options of full automatic test, full monitored test, selective tests, 12-hour central unit test and disk head cleaner.

The menu for the selective tests is extensive. The program checks your ROM, the RAM (six different ways), the video memory and display, and the printer functions.

If you don't have the time to sit and run through the tests, no problem; the Automatic Test mode can test your machine out for you with no one in attendance. It can then record the results to a line printer, disk drive, cassette or video display.

The Full Monitor Test is designed for those who wish to attend the machine while the program performs all the functions in a logical sequence. The Selective Tests allow you to perform any one of the tests designated on the testing menu. The Twelve Hour Test goes through the automatic sequence three times and records the results to cassette or disk.

All in all, the TBS packages are some of the finest I've run on my machine. I feel it's software like this that turns my TRS-80 from a rather expensive hobby or toy into a functional tool with a lot of power and versatility.

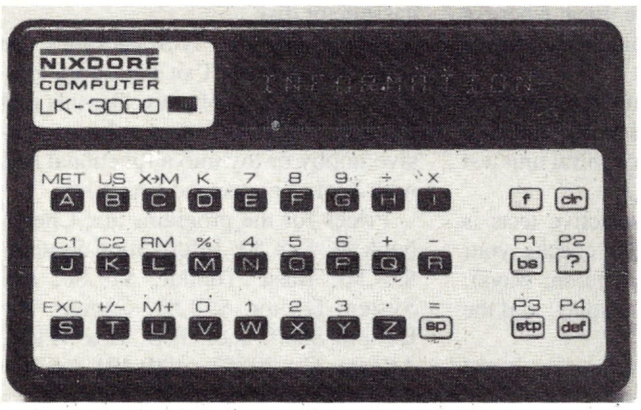
Prices for the programs are: Checkbook II, $18.50; Information System, $24,50; Basic Toolkit, $19.80; and System Doctor, $28.50. Contact The Bottom Shelf, Inc., P.O. Box 49104, Atlanta, GA 30357; (409) 491-7567.

WHAT'S COMING UP

SYSTEMS

Pocket-Sized Computer

Nixdorf Computer enters the market of portable computers with the LK 3000. The pocket-sized computer offers a range of personal and business capabilities. It can be programmed for use as a personal date book or telephone directory, or, in a business environment, it can function as a mini-terminal or data collection system capable of communication with other data processing systems via data communications interfaces.



The LK 3000 has application modules for use as a hand-held language dictionary to translate German, English, French, Greek, Italian, Spanish, Polish and Swedish words and idioms. The program library for the LK 3000 includes personal information modules and personal programming modules.

The LK 3000 has a suggested retail price of $140 with modules ranging from $35 to $150. For more information contact Nixdorf Computer Corporation, 168 Middlesex Tpke., Burlington, MA 01803; (617) 273-0480.
Circle No. 101

Computer-Based Toll-Free Answering Service

Voicegram, a nationwide toll-free computer-based answering service, is available to subscribers of The Source. Businessmen and others can send or receive electronic mail messages when a terminal is not available. You can dial a toll-free number anywhere in the continental U.S. and dictate messages by telephone. Messages can be retrieved from any terminal, communicating word processor or personal computer in the office or home, using a nationwide communications network. Subscribers can also call in to receive Voicegram messages sent to them.

Voicegram messages can be up to 100 words long and can be sent Monday through Friday, 9 a.m. to 9 p.m. Voicegrams are entered into the computer and forwarded to addresses within an average of 15 minutes. The charge for each message is $1.25 plus regular connect time charges of five cents a minute (25 cents before 6 p.m.). For multiple addresses, the charge is 75 cents for each additional address.

The Source is a computer-based electronic message and information system. It allows you to send messages over computer terminals via a nationwide packet switching network.

Users of The Source also have access to the UPI newswire, stockmarket and business news, New York Times consumer data base and daily news summary, educational programs, financial planning, and travel and discount shopping services.

To use The Source, subscribers need a personal computer or terminal. The cost of the service is $100 initial registration fee plus $2.75 an hour ($15 an hour during business hours). For more information contact Telecomputing Corporation of America, 1616 Anderson Road, McLean, Virginia 22102; (703) 821-6660. *Circle No. 102*

Executive Setting for TRS-80

ProData, Inc. announced an executive model computer package including a custom designed teak rolltop desk.

Two versions are available, each built around the TRS-80 microcomputer. Companion I has 16K memory, a video monitor and keyboard, a single mini-floppy drive, a cassette recorder and an electrostatic printer. Companion II substitutes an impact dot matrix printer and adds a second mini-floppy and an additional 16K of memory. A library of books, manuals, programs, supplies, (paper, diskettes and cassettes) and step by step instructions are included.

Computer components are encased in the desk with storage for accessories and room for expansion. Sources for additional equipment are also provided. Maintenance of all computer components is available from Radio Shack's national network of service centers.

Planned for the executive who doesn't have the time or inclination to assemble a variety of components, the package is designed to be complete and ready to operate.

These packages are priced At $3495 and $4995 FOB Fort Worth, TX, and are ready for immediate delivery. A 50% deposit reserves a place in the first run delivery. For more information contact ProData, Inc., 98-1122 Kahapili Street, Aiea, HI 96701; (808) 488-7104. *Circle No. 103*

PERIPHERALS

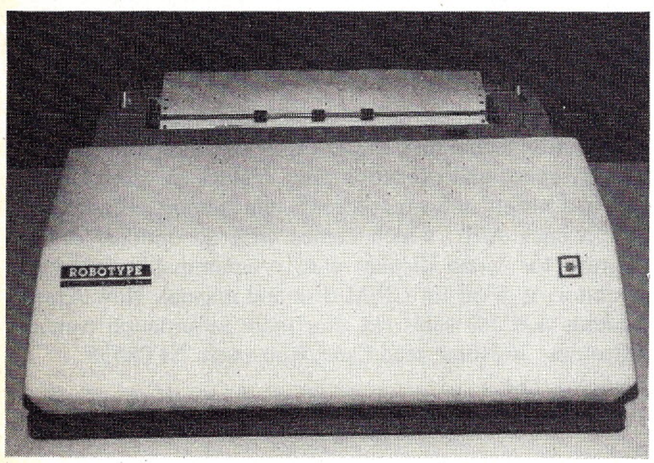
Unit Converts Typewriter to Printer

A device designed to convert typewriters into economic printers is offered by Applied Computer Systems, Inc.

The new system, the Robotype Model 2100, connects to any computer to obtain a printer for business, educational or home applications.

The unit can use a parallel interface (Centronics compatible), RS-232-C serial (with or without modems), 20MA current loop or TTL. The RS-232-C serial interface has 110, 134.5 or 150 switch selectable baud rates. Faster rates are available with a buffer option.

Robotype accomodates a variety of typewriter models including the IBM Selectric, Remmington Rand, Olympia and Facit. The unit, attached to a typewriter in a matter of minutes with no modifications to the typewriter, is placed over the keys of the typewriter. Plungers rest on the keys and depress the keys on command from computer input.



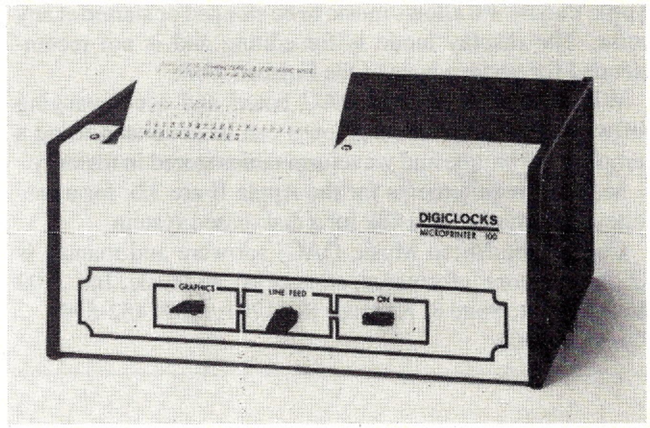
Robotype comes with all mechanical and electronic circuitry. The unit is housed in a fiberglass cover and types at the maximum speed of the typewriter in use.

Several options enhance the capabilities of the Robotype, the company said. If a keyboad is added, the unit becomes an input/output terminal. Through the serial interface, Robotype can communicate with any computer, remotely or locally, and the addition of a microprocessor allows the unit to stand alone as a computer system with a working space of 2000 characters to program in integer BASIC.

The Robotype, covered by two patents, is warranted for parts and labor. The unit is available from stock to 30 days for under $1000. For more information contact Applied Computer Systems, Inc., 77 E. Wilson Bridge Rd., Worthington, OH 43085. *Circle No. 104*

Low-Cost Printer for Pet

Digiclocks is offering a low-cost microprinter for the Pet. The Microprinter 100 does the entire Pet screen character set in both positive and negative writing. The unit is supplied



complete and ready to operate with the Pet; no extras are required. It is small, reliable and addressable, the company said.

The unit is available at an introductory price of $390. For more information contact Digiclocks, 3016 Oceanview, Orange, CA 92665; (714) 557-5252. *Circle No. 105*

Music System for the Apple II

Micro Music Inc.'s Micro Composer, an Apple II compatible music system, leads you through entering, displaying, editing and playing music with up to four voices with a four octave range.

Micro Composer lets you:

- See all four voices at the same time the music is played.
- Enter music by a coding system which keeps track of rhythmic durations.
- Program pitch, rhythm and timbre. Tempo is varied by the Apple II paddle.
- Choose from seven different tone colors for each voice, or create your own tone color.
- Save your music on disk or cassette.

Micro Composer comes with an instruction manual, software disk (or cassette) and a Micro Music DAC music card. The DAC plugs into the Apple II extension slot and is connected to an 8 ohm speaker.

A 4-octave range consists of 25 equal-tempered chromatic steps. Rhythmic durations are whole, half, quarter, eighth, sixteenth and thirty-second notes and their dotted values.

Music with 1, 2, 3 or 4 polyphonic voices may be entered in vertical sound sets. Each set contains all of the rhythm and pitch information needed during that time unit. With 4 voice music, 744 sets may be stored in the 8K memory reserved for music.

Micro Composer has 7 pre-set timbres: rich string, wind-flute, brass, oboe-bassoon, clarinet, electronic organ and funky oboe. There is capability for constructing and saving your own timbres. The timbres are limited to 16 harmonics and restricted to timbres below 5 KHz.

The music system, command driven, has available Play, Display (and play), Compose, Edit, Fourier, Voices, Meter, Timbre, Read, Write, Help and Quit. Editing is possible for any set. In the edit mode, the stored notation set is displayed as well as the new notation being entered. The button on the

Apple Paddle #0 allows music to be stopped or started at any point. The display mode is for editing and is not recommended for performance of the finished work.

With the Micro Music DAC board and accompanying software, four simultaneous tones can be generated using a sampling technique and waveform tables stored in memory.

Memory requirements for the Apple II are 32k for a cassette-based system and 48k for a disk-based system.

Cost of the Micro Music DAC, software and manual is $220. For more information contact Micro Music, Inc., 309 W. Beaufort, Suite 8, Normal, IL 61761; (309) 452-6991. *Circle No. 106*

Printer for TRS-80

Matchless Systems has added a printer to its line of TRS-80 related products.

The 80-column, bidirectional, 5 × 7 dot matrix printer utilizes a print mechanism of simple design and high reliability, the company side. The print head has a life of 100 million characters.

Among the other features are a print speed of 125 cps and a throughput print speed of 63 lpm. The adjustable sprocket feed mechanism allows use of forms from 4-1/2″ to 9-1/2″ wide, with loading from either the bottom or rear. A full 96 ASCII set permits printing upper and lower case characters which can be expanded for double-width fonts in bold face. A Vertical Format Unit provides pre-programmed/programmable tab positions, top of form and bottom of form.

In addition to its use with the TRS-80, the printer's Centronics-compatible parallel interface makes it possible to use directly with other computers like Apple, Sorcerer or any standard computer with parallel interface.

While the printer sells for $777, Matchless has a special introductory price of $749. For more information contact Matchless Systems, Dept. 3, 18444 South Broadway, Gardena, CA 90248; (213) 327-1010. *Circle No. 143*

SOFTWARE

Software Turns TRS-80 into Printing Calculator

You can make your TRS-80 (Level II, 16K) into a printing calculator with Manhattan Software's Calculator Plus.

The program, an on-screen calculator if you don't have a printer, provides chain and mixed calculations on screen with a print command to record intermediate steps, if desired, and the final answer. Entries in long add-and-subtract operations can be checked with an on-screen review command, or printed out for verification and a permanent record. The program works with printers down to Quick Printer II size.

Significant figures can be retained in a separate memory section, with titles entered by the user for identification. Memories can be printed out as a table, or any memory can be carried to a calculation mode as a constant.

An optional dollar format is provided both on-screen and in printouts, and double-precision is used throughout for accuracy of calculations.

The program is priced at $9.95. For more information contact Manhattan Software, Inc., P.O. Box 5200, Grand Central Station, New York, NY 10017; (212) 427-4718. *Circle No. 107*

APL for the 8080/8085 and Z-80

Softronics' APL, an interactive general-purpose programming language, is available for 8080, 8085 and Z-80 microprocessors. The company said its APL has most of the functions and operators of full APL, including n-dimensional inner and outer product, reduction, compression, general transpose, reversal, take, drop, execute and format, system functions and variables, and system commands.

APL runs under the CP/M operating system, residing in 30K bytes of memory. It is "ready-to-go" in ASCII, using CP/M standard I/O. The interpreter runs in a variety of character set configurations. In addition to the standard ASCII mnemonic representations, it supports typewriter and bit-pairing ASCII-APL character sets. It can run with user-supplied I/O drivers. The shared variable mechanism allows CP/M disk I/O.

Abrams' descriptor calculus and shared data storage are optimization techniques employed by the interpreter to save memory space and execution time. Values are stored internally in a variety of formats for efficient memory utilization.

Softronics APL, which comes with an optional driver program for video display with programmable character generator, is $350 for CP/M disk and manual. New Jersey residents add 5% sales tax. For more information contact Softronics, 36 Homestead Lane, Roosevelt, NJ 08555. *Circle No. 114*

Software for North Star Systems

Generating reports or preparing for a real estate exam with a North Star System is possible with releases from Media 2001.

RPTGEN is an applications package designed to render up to 18 fields of data/comments in an organized report complete with headings and sub-headings. Each field can be specified to a particular use. Formats already in use include Property Listings — Real Estate; Client Data — Real Estate; Mortgage Loan Applications — Financial Institutions; Client History — Medical, Dental, Psychiatric; Product Specifications — Manufacturing; Library Summary — Reference.

Many other uses are available through RPTGEN. The only limitation for applications is the user's imagination, the company said. Each field can be used to store such information as name, address, telephone, physical attributes and descriptions, account status or comments.

RPTGEN offers flexibility in handling a report. The following sub-programs are standard in the package: Add, Change, Delete, Directory, Print1, Compact, Initiate.

Real Estate, a self-study program of twenty-one seg-

mented lessons for preparation to take the real estate agent's/broker's licensing examination, includes real estate law, finance, mathematics, appraisal techniques, public control and miscellaneous topics.

Utilizing a computer memory core of dialogue, each lesson allows the student to converse directly with the computer in an interactive exchange.

Each question reflects actual "exam" situations a student encounters when applying for a State of California real estate license. Although designed for the California examination, the course covers essentially all major areas of state examinations throughout the U.S., the company said.

The RPTGEN system, written in North Star BASIC with versions to be released for the TRS-80, Apple and Texas Instruments systems, retails for $399, which includes customization for the purchaser.

Real Estate is designed to run in North Star BASIC with 32K of RAM. Versions for the TRS-80, Apple and Texas Instruments systems will follow.

The study program retails for $299. For more information contact Media 2001, P.O. Box 614, Corte Madera, CA 94925; (415) 924-5311. *Circle No. 109*

Data Base Management System

SCDP's (Software Consultation, Design and Production) Vulcan Data Base Management system has 38 English language-like commands to manipulate files, records, fields and scratch-pad variables. Vulcan is an 8080 microcomputer implementation of a user oriented system used in industry and educational institutions. Vulcan, fully operational, has been field tested for over eight months, the company said.

Vulcan is written in 8080 assembly language and operates on 8080 or Z-80 systems under CP/M or PTDOS. It requires a minimum 36K CP/M and one or two disk drives.

Commands include: Sort, Report, Create, Insert, Append, Edit, Delete, Recall, Goto, Skip, Save, Store, Accept, Input, Set, Wait, If, Return, Cancel, Copy, Replace, Locate, Display, Do, List, Loop, Use, Count and Sum.

Structured data records can be selectively chosen for processing using complex Boolean, string or mathematical expressions. Functions and macro substitutions are also incorporated. Certain commands are entered directly. Other commands initiate a dialogue extract additional information.

The program is fault tolerant. Whenever the user enters erroneous input, Vulcan prompts the user to enter corrections.

Commands can be executed in either interactive or program mode. In the interactive mode, a command is executed as soon as it is entered. In the program mode, commands are entered into a file in structured form and then executed with a single input statement.

Vulcan can accept or store data in standard ASCII files to be compatible with BASIC, Fortran and Cobol.

The system is supplied on 8 or 5 inch (CP/M format) diskettes or 8 inch PTDOS diskette with manual for $490. The manual alone is $25. For more information contact Software Consultation, Design and Production, 6542 Greeley St., Tujunga, CA 91042; (213) 352-7701. *Circle No. 118*

Music on the Apple

Hayden's Song in the Key of Apple by Ira Lopatin allows your Apple II to play music. No programming background is necessary.

The program offers many choices. You can play The Stars and Stripes Forever, Wild Horsemen, Fanfare from William Tell Overture, or a song you have previously programmed and saved on tape. In additon, you can directly input and play out on command other music you create. Songs may be written with up to 200 notes, including rests, per musical piece. The program written in Integer BASIC, allows a two-octave range, playing all notes including sharps, flats, and rests. Multicolor graphics accompany all music, even that which you create.

The program sells for $10.95. For more information contact Hayden Book Company, Inc., 50 Essex St., Rochelle Park, NJ 07662; (201) 843-0550 *Circle No. 111*

Apple II Utility Routines

ISAM-DS by Decision Systems consists of an integrated set of 15 utility routines for an Apple II using Applesoft. The package facilitates the creation and manipulation of indexed files.

Utilizing the routines, records on indexed files may be retrieved randomly or in sequence. Each record is identified by a key data value. The key values do not have to be part of the record; they do not have to be unique for each record; and partial key values may be used in retrieving records.

The interface between ISAM-DS and an Applesoft program is through a single entry point (GOSUB) and nine variables. Files can be created, opened, closed, copied and erased. Records can be written, read, changed and deleted. File space that is freed by deleting a record is automatically reused when another record is added. There is never a need to "clean up" a file because of update activity.

The ISAM-DS package includes the set of routines, documentation for the routines and a mailing list program that demonstrates ISAM-DS capabilities. Append routines for DOS 3.1 and 3.2 are also included. The append routines are used to join the ISAM-DS package to an Applesoft program. Cost of the package, which requires Disk II is $50. For more information contact Decision System, P.O. Box 13006, Denton, TX 76203; (817) 382-6353. *Circle No. 113*

Sports Packages for Apple II, TRS-80

NSP, Inc., offers two software packages for sports enthusiasts: one computes and records golf scores and handicaps; the other is an interactive computer bowling game. Both packages are designed for the TRS-80 and Apple II.

Gold Handicap System uses standard USGA handicap formulas based on the proper number of lowest scores. The program retains the most recent 20 scores per player and retains the individual course rating.

The package displays and prints player names, scores and handicaps. You can delete players, correct names and/or scores, add new scores to existing scores, reject unreasonable scores and sort by player name.

Golf Handicap System handles 180 players on a 16K Level II TRS-80 or 500 names on a 32K Apple II with Integer Basic. The TRS-80 version is expandable to 500 names with additional memory.

With Computer Bowling, a player may play a single game or two players may compete. Players, identified by name, can select duckpins or tenpins.

The game features left- or right-hand, straight or curved, fast or slow ball deliveries; running displays of scores and marks; realistic pin display; random break selections dictated by the ball's contact with a pin; and random ball path bias to make every game and every ball delivery different.

The Apple II version, which executes from diskette or cassette, offers noise-accented pin action.

Golf Handicap System consists of two programs on a single cassette for $85 and will execute from cassette or Apple diskette. Computer Bowling is priced at $9.95. For more information contact NSP, Inc., P.O. Box 3092, Crofton, MD 21114; (301) 721-3849. *Circle No. 117*

Explore Dungeons with TRS-80

Software Exchange announced Dungeon Explorer, an adventure game for the 16K TRS-80 Level II computer.

Dungeon Explorer is a single player game of combat and adventure, based upon Dungeons and Dragons. Deep underground in the Dungeon of Xanadu, there are incredible treasures. But these treasures are guarded by vicious monsters. The object is to survive the dangers and become a legendary super-hero. Each trip into the dungeon is different. The game can last for hours, the company said.

Dungeon Explorer is available for $8.50 including cassette tape and instructions. For more information contact Software Exchange, 2681 Peterboro, W. Bloomfield, MI 48033. *Circle No. 125*

Subroutines in North Star BASIC

HELPB5 is a collection of subroutines in North Star BASIC which use dynamic memory assignment to do list processing. Available subroutines include Create, Destroy, File First, File Last, File Ranked, Remove First, Remove Last and Remove, plus additional debugging subroutines.

You can create sequential sets of array elements which describe objects. By filing sets on and removing sets from various lists, complex processes can be simulated. Object sizes and number of lists, set by the user, is limited only by available core.

File First and File Last place an entity first or last on a list; File Ranked places an entity on a sorted list based on the value of a selected array element. Remove First, Remove Last and Remove perform the opposite functions. The debugging routines allow you to print all the objects on a list, determine if a particular object is on a specified list, if an array element has a specified value and print the array elements for a particular entity.

HELPB5 needs a minimum of 32K RAM. A user's manual and sample simulation program are provided. The price is $48. For more information contact American Planning Corp., 4600 Duke Street, Suite 425, Alexandria, VA 22304; (703) 751-2574. *Circle No. 126*

Art and Math for Imagination Machine

Two new cassettes from APF Electronics turn the Imagination Machine into an artist's tool or math tutor.

Artist and Easel lets you be a creative artist as The Imagination Machine becomes a paint brush, palette and designing pen to draw, doodle and play. Cassette storage lets the artist bring back past creations on cassette tape.

APF calls the Artist and Easel cassette an educational game which teaches children shapes, sizes, dimensions and color perception, while they play and develop cartoons and a variety of computer configurations.

Math Tutor performs like a private tutor as it directs you on the basis of ability and learning speed.

As you begin to use the math cassette, The Imagination Machine automatically determines the level of ease or difficulty you are at. By varying problems and arithmetic questions, Math Tutor makes it fun and easy for you to gain speed, knowledge and dexterity in math, the company said.

If you do well at one level in the program, the Personal Performance Response feature offers more challenging problems to develop skills. If additional reinforcement is needed, the computer compensates with easier problems until you are ready to try more difficult concepts.

Artist and Easel and Math Tutor sell for $29.95 each. For more information contact APF Electronics Inc., 444 Madison Ave., New York, NY 10022; (212) 758-7550.
Circle No. 122

Application Packages for Pet

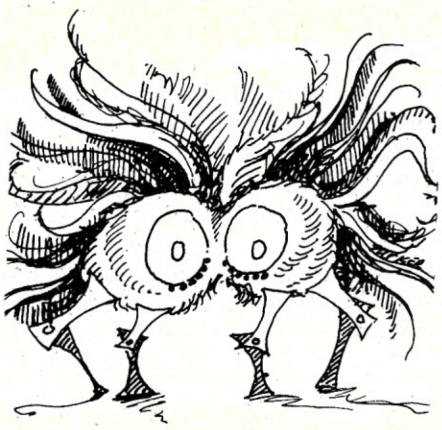
Three software packages from Total Information Services provide practical applications for your Pet at home and at your business.

A Checkbook Program has checkbook and budget applications. It assists in balancing a checkbook; selects and displays checks by person, purpose, or date; and sums checks by category or person. An Accounts Program allows you to create a data base for company names, addresses, invoice and purchase order numbers, and amounts of purchase. It locates information on specific companies, determines amounts owed and displays past-due accounts. The Calendar Program keeps track of appointments in an office or your schedule of social engagements. It is an all-purpose datebook complete with events, dates, times and notes. TIS also offers a Mailing List Program and a Micro Text Editor.

All software packages include a cassette copy of the source program, a source listing and an instructions manual.

Each of the five TIS software packages costs $9.90 for the cassette version or $12.95 for disk version. For more information contact Total Information Services, Box 921, Los Alamos, NM 87544; (505) 455-7049. *Circle No. 121*

Science Fiction



Turn your daydreams and visions of the future into cash. We're looking for short (500 to 2000 words) science fiction stories dealing with the future of microcomputers – their possible uses and their roles in society and in people's lives. Stories must be original and not published elsewhere. Submit your typed, double-spaced manuscript to Personal Computing, 1050 Commonwealth Ave., Boston, MA 02215.

Share Your Home Programs

How do *you* use your computer at home? Family finances? Budgets? Meal planning? Entertainment? Teaching the kids? Word processing? Home security? Investment planning? Helping with your *other* hobby?

Our readers are as software-hungry as you. So why not share the home applications programs you've developed? Send us an article describing your application and the program you wrote to implement it. Be sure to include a program listing and sample run.

Remember, readers aren't familiar with your program. So explain in detail what the program does and how it does it. Include here the overall structure of your program as well as any special algorithms or routines you've used. Give suggestions for modifying or expanding the program for other applications or other situations.

All submissions should be original, typed (not all CAPS), double-spaced and neat. Include your name and address on the first page of the article and enclose a self-addressed, stamped envelope for return of material. Also, please use a fresh ribbon on our printer for program listings and sample runs.

Feel free to call us at (617) 232-5470 if you have any questions or want to discuss specific article ideas.

Mail your manuscript to: Editor
Personal Computing
1050 Commonwealth Ave.
Boston, MA 02215

CP/M Software Systems

H & H Associates, Inc., markets eleven software packages for systems operating under CP/M. Hardware and software requirements for the packages include a minimum of 32K memory, CBASIC/CRUN Version 1, QSORT, two disk drives and a 132-column printer.

According to the company, all the systems are easily modified. Code is structured and all control information is maintained in a common file. Some of the control information common to most packages are clear screen command, current computer date, printer form feed commands, files names, record lengths, user name and system name, and version.

All record layouts are defined in a common source code member and each program utilizing the file copies the appropriate member, eliminating the need to change every program that accesses the file when changes to record layout are made.

Error messages include a unique number which may be used to find a complete error message description and valid responses and/or actions in an error message index.

All systems are supplied with a user's manual and source code in printed format. Source code is provided on one or more single density 8-inch CP/M diskettes for an additional charge of $50. All orders must be accompanied by check or money order and are shipped by UPS within the U.S.

Packages available are Accounts Receivable, $200; Apartment Management, $200; Client Write Up, $100; Law Office Billing $200; Order Entry, $200; Medical Office Billing, $200; Categorical Information System, $100; Payroll, $100; Statistical Analysis, $150; Inventory, $200; and Automated Forms Control System, $75.

For more information contact H & H Associates, Inc., P.O. Box 2663, Renton, WA, 98055. *Circle No. 115*

Income Property Management Software

A-T Enterprises announced software designed for the broker/property manager and the professional property manager. The system provides management and accounting control for income properties including apartments, condominiums, houses, mini-warehouses, mobile home parks, office buildings and shopping centers.

The software keeps track of all income and expenses, generating management statements, reports and exception reports necessary to control and manage income properties, the company said. The chart of accounts and operating statement (profit and loss) can be customized to duplicate your current method of doing business.

Operating statements shows month and year to date income and expenses with a percentage comparison of expenses to income. Budgeting, list of tenants, delinquency reports and vendor information with year to date payments are also incuded. The system can handle partial, late charges, over and advance payments. A complete audit trail is provided.

The system is interactive with prompting questions displayed on the CRT. All information from the system can first be viewed on the screen and then printed at your option.

Property Management also includes business and real estate related programs like income property analysis, loan payments, loan amortization, ballon payments, rate of return and depreciation schedules.

The software operates on any 8080 or Z-80 microcomputer with 48K of memory, dual 8″ diskette drives, a CRT and 80-column printer. The software, written in CBASIC, runs under the CP/M operating system. The approximate capacity of the system is about 75 properties and 1500 units.

The property management software costs $750. A demonstration diskette with manual is $95 and can later be applied toward the total software price. Credit cards and COD orders are accepted. For more information contact A-T Enterprises, 221 No. Lois St., La Habra, CA 90631; (213) 947-2762. *Circle No. 129*

Programming Aids for Apple II

Dakin5 Corporation, developer of The Controller business system for Apple Computer, is releasing the *Dakin5 Programming Aids,* a package of seven programming aids for the Apple II.

The first of the programming aids, The Lister, sends BASIC programs to the printer where they are listed using the full line capacity of the printer. The Peeker displays or prints all or selected records from a text file. The Cruncher removes REM statements and compresses code in Applesoft programs. The Diskette Copy, a diskette-to-diskette copy, also verifies that the output matches the input data. The other aids are The Prompter, The Calculator and the Text File Copy.

For a demonstration of the *Dakin5 Programming Aids,* visit an Apple dealer. Suggested retail price is $39.95. Over 30 pages of documentation are included with the package. For more information contact Dakin5 Corporation, 7475 Dakin St., Suite 507, Denver, CO; (303) 426-6090.
Circle No. 112

TRS-80 Automatic Phone Dialer/Timer

Blechman Enterprises's Telephone Dialer Program for the TRS-80 Level II, written in BASIC, will hold up to 500 names and telephone numbers in a 16K memory or about 30 names and numbers with a 4K memory.

You may select instructions on how to enter names and numbers in program DATA statements. You can then request a list of all names in memory, which will be displayed in alphabetical order on the screen, and enter the desired name. The computer displays the name, number and area code, then displays each digit as it "dials." The dialing speed, fast or slow, is accomplished through a simple external telephone interface circuit consisting of a Radio Shack $3 relay, SPDT switch, LED, resistor and 9V battery. The relay contacts are connected in series with the telephone line. The system can be used with all telephone circuits, rotary or tone.

No modification is required to your TRS-80. The cable that normally goes to the cassette remote jack is used to pulse the interface relay. The switch allows normal computer cas-

sette operation, or dialing. The LED and resistor monitor status and dialing.

After the program dials the desired number, press Enter when the party at the other end picks up the phone; the computer displays the length of the call continuously by the second. At the end of the call you can insert the telephone rates for the call and the computer displays the total call time and charge.

The program is furnished on cassette with a schematic and parts list. The "flip-side" of the cassette has the Phone Toll-Charge Program which keeps continuous track of the time and charges of any call without connection to the phone. Price for the program is $10 including shipping. California residents add 6%. For more information contact Blechman Enterprises, 7217 Bernadine Ave., Canoga Park, CA 91307; (213) 346-7024. *Circle No. 116*

Software for Home and School

Educational and home programs from Carta Associates include VIC (Visual Instructional Computer) and the Carta Lesson Library series. VIC uses the TRS-80 display to teach basic computer architecture plus assembly and machine languages to the casual or beginning student. The Carta Lesson Library, a three-part package of study and test materials, can ask questions in a variety of formats and, at the instructor's discretion, in timed test sequences. The Lesson Library's components are the Master Lesson Programs, the Lesson Tapes and the Lesson Builder, all of which are sold separately. The Lesson Tapes provide study and review data in a variety of subject areas.

All programs require a Level II 16K TRS-80. VIC costs $19.95. In the Lesson Library, the Master Lesson Programs costs $29.95 (which includes one Lesson Tape of your choice). The Lesson Tapes costs $7.95 each. The Lesson Builder, which lets you create your own lessons, costs $39.95. All are available through computer stores.

For more information contact Carta Associates, Inc., Education Products Division, 640 Lancaster Ave., Frazer PA 19355; (215) 647-9600. *Circle No. 133*

Disk Drive Timer for TRS-80 and Apple II

Disco-Tech has introduced DDT, a disk drive timing program for both the TRS-80 and Apple II. DDT lets you keep track of drive motor speed on a routine basis and adjust it if it's running fast or slow. Improper speed causes data loss and incompatibility among diskettes. DDT reduces down time and costly repairs, the company said.

The program works on any disk drive and provides a real-time graphic display of motor speed on the video screen, which lets you analyze the speed of each drive and adjust the speed within one-tenth of one rpm, out of an optimum 300 rpm.

You don't have to be an electronics technician or a mechanic to use DDT, the company said. All you need is two screwdrivers, the DDT program and five minutes. DDT comes with documentation which takes you step-by-step through the analysis and adjustment procedure.

The DDT manual for the TRS-80 contains specific instructions for adjusting both new and old model Radio Shack drives, Shugart, MPI, Pertec and Vista. Apple drives are treated in the Apple version of the program.

DDT comes in two versions, for the TRS-80 and for Apple II. The TRS-80 DDT program may be purchased on cassette at $14.95 retail or on diskette at $19.95. The Apple version of DDT is available on diskette only at $19.95. For more information contact Disco-Tech, P.O. Box 11129, Santa Rosa, CA 95406; (707) 523-1600. *Circle No. 130*

Telecommunications Facility for CP/M

Lifeboat Associates markets the Byrom Software Telecommunications Access Method (BSTAM). This system enables transmission of program or data files between any two computers on whch BSTAM is installed, and is compatible with all 8080/Z-80 systems using CP/M operating systems or a derivative, including Heath and TRS-80 adaptations.

Transmissions are made over a normal voice grade telephone line at 300 baud and over direct wire interconnections at 9600 baud. Other than error checking and protocol information, no data expansion is performed, resulting in fast transfers. BSTAM is a commercially oriented telecommunications facility with many "big computer" features built into it, such as CRC error checking and group file transmission.

BSTAM can precisely transfer data over poor circuits, with automatic retry provisions and perfect reporting in the event of "hard" errors. The user interface provides for expansion of file names, permitting a sequence of files up to a whole diskette to be sent, the sending module automatically announcing the file names to the receiving computer.

BSTAM is supplied on diskette in all popular formats. The license fee of $150 covers installation on the single system only. The documentation is available separately at a price of $5. For more information contact Lifeboat Associates, 2248 Broadway, New York, NY 10024; *Circle No. 134*

Software for 32K North Star System

Denron Amusements has developed software for the North Star system to include a game series and home applications.

Games software include Panzer, Blitzkrieg, Fall of the Third Reich, D-Day, Armorcar, Porkchop Hill, Africa Corps, Waterloo, The Battle of Monmouth, Starship Troopers, Middle Earth, Invasion of the Mud People and The Boston Marathon. Home applications developed to date are Magazine Article System, Home Budget System, Buy-Used-Car System and Learn Arithmetic Program.

All programs were developed with Release 4, require one drive, and utilize a 32K system.

All games are currently priced at $9.95 plus $1 mailing charge. (NJ residents must add 5% sales tax). The Learn Arithmetic program sells for $5.95 and all other home application systems sell for $14.95. All prices incude the cost of the disk. For more information contact Denron Amusements, 13 MacFarlan Avenue, Hawthorne, NJ 07506. *Circle No. 131*

32K Structured Basic for Cromemco

Cromemco 32K Structured BASIC, designed to run in 64K Cromemco systems, combines the advantages of a modular language with the ease of programming in BASIC, the company said.

Structured BASIC assists a programmer in building a program from logical blocks of code. The high level language facilitates program development, leads to a more readable code and reduces debugging and maintenance of programs.

Structured BASIC, an extension of Cromemco 16K BASIC, contains all features of 16K BASIC plus long variable names, statement labels, an in-line BASIC editor, a sophisticated KSAM, Procedures, and control structures including If-Then-Else, While-Endwhile and Repeat-Until.

Variable names may contain up to 31 characters, facilitating program debugging and maintenance.

Descriptive statement labels may be used in place of statement numbers to reference lines in a BASIC program.

The line oriented text editor enables the user to delete or insert selected characters in a statement without retyping the entire line.

Incorporated in Structured BASIC is KSAM (Keyed Sequential Access Method), a disk storage and retrieval system. KSAM stores a data file based on the contents of a record rather than on an arbitrary record number. A record can be retrieved by specifying the contents of a key field.

Procedures, which allow for modular programming, are called from memory or a Procedure library and may include both global and local variables. Procedure names may contain up to 31 characters.

Cromemco 32K Structured BASIC is available for use on Cromemco systems for $295 on either 8″ disk (Model STB-L) or 5″ disk (Model STB-S). For more information contact Cromemco, Inc., 280 Bernardo Avenue, Mountain View, CA 94043; (415) 964-7400. *Circle No. 138*

Business Mailing Systems for TRS-80

TBS Inc.'s Business Mailing System is designed for large scale business users with a TRS-80, at least 32K of memory, a printer and two disk drives.

The system allows you to store up to 150,000 names on a single large file composed of multiple diskettes. The program sorts entries into Zip code order and alphabetical order within the Zip code. As new entries are made, the file is expanded automatically. In other words, the file will grow from one diskette to two, all the way up to 300 diskettes. Each diskette holds 500 names, the company said.

The system allows you to use one through four labels at your discretion. It provides for the printing of either three- or four-line addresses. The mail list further allows you to program which names you wish to print out by the use of up to ten exclusive and non-exclusive codes.

This system is for the business user. If less than 1000 names are to be used, other systems would be less expensive and more versatile, the company said. For large scale mailing lists, however, this system provides a convenient way to do large scale processing on a small computer, the firm added.

Price for The Business Mailing System is $125. For more information contact The Bottom Shelf, Inc., P.O. Box 49104, Atlanta, GA 30359; (404) 939-6031.
Circle No. 147

Mini Disk Software for Apple

A series of six mini disk application programs is available for Apple II computers from Williamsville Publishing Company. The application programs require an Apple II or Apple II Plus computer with 32K RAM memory, one Disk II drive, and Applesoft II in ROM (firmware card). All programs come on floppy mini disks.

Disk A1, "Book Library," provides a disk-based catalog for inputting over 1000 selections cross-referenced by author, title and subject. You can store and search more than one subject per title. This program can also be used for magazine articles, software library or catalog data.

Disk A2, "Record Library," is similar to "Book Library" but is designed for music tapes and record collections. Data is cross-referenced by composer, title and performance. More than one performance can be listed per title. As with Disk A1, catalog data may also be stored.

Disk A3, "Malum II . . . Imperial Roman Programmable Computer By Command of Caesar," is a Latin language based interpreter similar to BASIC. It takes the Latin language equivalent of BASIC commands. All numeric input and output is via Roman numerals. Two sample programs and an 11-page documentation manual comes with the disk.

Disk A4, "Graphics Games," has Bowling, Checkers, The Derby and Tic-Tac-Toe. The feature game program is The Derby, a horse race in high resolution graphics. Win, place and show betting is allowed for up to six players. For each race, seven horses are randomly selected from a stable of 21. The track record of each horse is updated on the disk file after each race. The game can last for ages or you can start the record over at any time. You can name any or all the horses. The other three games use Apple's low resolution color graphics.

Disk A5's "Checkbook Program" can store up to 62 checks and deposits per month (32K RAM) or 244 per month (48K RAM). The disk can be used continuously for 48 months or indefinately by deleting old files. Entries can be searched on payee, purpose, user-defined code, check number, or combinations of payee and purpose, or payee and code. All entries found by search are totalled. The program is designed for both home and small business.

Disk A6, "Page Processor," is a word processor. Originally designed for the Centronics 779 and Apple, the program is easily modified for use with any popular printer and can also be used as a CRT screen word processor without a printer. Features include editing capabilities such as line centering, tabbing, line insertion and deletion. Each page is automatically stored on disk for future reference.

Individual mini disks with documentation sell for $19.95. For more information contact Williamsville Publishing Company, Box 250, Fredonia, New York 14063.
Circle No. 123

Protection for BASIC Programs

Data Associates has released a program that automatically protects BASIC programs written for the TRS-80. UNLIST8 runs on a single disk system with 32K memory. BASIC programs are protected against unauthorized modification and are made confidential. Hidden passwords and copyright notices selected by the user are inserted and then the program is converted so that it cannot be listed or printed. The protected program can still be RUN, CSAVED, CLOADED, DISK LOADED and DISK SAVED as usual.

Options permit unlisting all lines, each nth line or specified blocks of line numbers. This program can also be used to unprotect or relist a protected program provided you know the program's password. It can relist each line, or blocks of specified line numbers. Certain lines can be left unprotected.

UNLIST8 converts any named BASIC program (saved in ASCII form on disk) into the protected version and automatically saves it on the same disk under a new name selected by the user. After the conversion is complete, a summary is displayed of the original name, the converted name and the elapsed time, as well as a count of both the input and output bytes and lines. The program displays prompting messages as needed.

An instruction manual and three copies on cassette are provided. Cost is $19.95 postpaid. For more information contact Data Associates, Box 882, Framingham, MA 01701.
Circle No. 146

Business Software for TRS-80 Model II

Micro Architect has converted its present software products to run under the new TRS-80 Model II computer; the programs have been modified to make use of the expanded hardware capabilities such as a bigger screen, more memory and disk storage.

The company guarantees all Model II programs and will provide bug fixes and enhancements. The president of the company writes, documents and supports all the Model II packages.

A company representative said several levels of support are provided to the user: pre-sale support includes manuals for $10 for examination which state all the known limits of the package; after-sale support includes a free newsletter for one year. The newsletter describes operation procedures for your Model II. Software enhancements and bug fixes will also be described in detail.

Long-term support allows full credit towards purchase of related, completely revised package, the company stated. Software improvements will keep up with hardware enhancements, such as hard disks or intelligent printers. For those who lose track of updates in the newsletter, revised versions can be obtained on diskette for $15.

Initial offerings include advanced mailing list system at $99, a word processor at $49, accounts receivable at $149, inventory system at $149 and a data base manager without user programming at $199.

All packages are integrated but can run stand-alone. Prices include 8" diskette, documentation, postage and newsletter

for one year. A 10% discount is allowed for orders of more than one package.

For more information contact Micro Architect, 96 Dothan St., Arlington, MA 02174.
Circle No. 124

COMPLEMENTS

Cassette Accessories for Erasing, Rewinding

Magnesonics Sales offers two accessories for your computer cassette recorder: Erase-Sure and Rapid Rewind.

Erase-Sure, a compact, self-contained erasing unit, utilizes a patented principle that consists of erasing prerecorded magnetic tape by passing it through a rotating magnetic field.

Rapid Rewind, also a compact unit, will rewind a C-60 cassette in approximately 30 seconds, according to the company. The firm said the accessory will also stabilize tape tension, eliminate tape binding, help control wow and flutter, save wear on recording equipment and achieve uniform tape packing.

Both units sell for $24.50 plus $1.50 shipping and include 4 "AA" batteries. For more information contact Magnesonics Manufacturing and Sales, P.O. Box 758, Ventura, CA 93001; (805) 642-3092. *Circle No. 149*

Lowercase Modification Kit for TRS-80

Emmanuel B. Garcia Jr. and Associates offers a lowercase keyboard modification kit for the TRS-80.

The kit comes with wire, solder, control key, 2102 memory chip, slide switch, mounting hardware and documentation. Documentation begins with instructions for static-proofing your workbench and ends with an explanation of how the modification works. Eight illustrations and diagrams aid installation.

The 2102 chip is connected to a slide switch allowing the TRS-80 to be used with or without lowercase. To minimize the chance of damage to the chip, the wires harnessing chip and switch have been pre-assembled, the company said. The control key has gold plated contacts for long life.

The kit sells for $19.95. For more information contact Emmanuel B. Garcia Jr. and Associates, 3950 N. Lake Shore Dr., Rm. 2310, Chicago, IL 60613; (312) 348-6562.
Circle No. 139

Got an unusual application?

If you use your computer for an interesting, intriguing or unusual application (or know someone who does), our readers would like to hear about it. Why not write up a short (500 to 1000 words), original article telling us about it? Make it light and newsy, and include black-and-white photos if appropriate. Send your submission to Random Access, Personal Computing, 1050 Commonwealth Ave., Boston, MA 02215.

Labels for Magnetic Media

Pressure-sensitive labels specifically designed for magnetic media are available from the Consumer Products Division of Dennison.

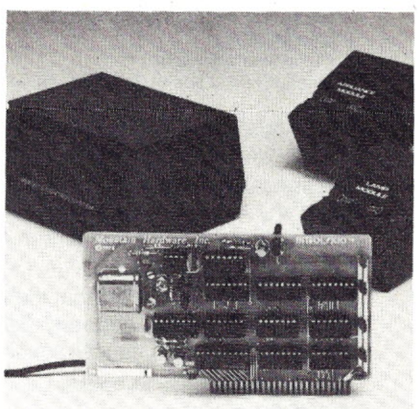
The labels aid identifying, coding, filing and retrieving cassettes, magnetic cards and floppy disks. They come in standard shapes and sizes and are color coded. Self-adhesive, the labels are removable to permit easy replacement. The cassette labels have die-cut windows to show the tape length and keep the spools free running. The new labels fit right over the original identification area on the media.

Cost per 100 labels for cassettes, 5-1/4″ diskettes and 8″ diskettes is $1.95. For more information contact Consumer Products Division, Dennison Manufacturing Company, 300 Howard Street, Framingham, MA 01701; (617) 890-6350. *Circle No. 120*

P.C. BOARDS

Remote Control through an Apple

Introl/X-10 from Mountain Hardware allows you to remotely control 110 volt AC devices by commands sent over existing building wiring. Utilizing an Apple, command signals are sent to a BSR/System X-10 Command Console. The console then sends signals over existing wiring to control electrical devices.



The Introl Controller board plugs into a peripheral slot of the Apple. It transmits control signals with an ultrasonic transducer to the BSR/X-10 Command Console, which may be plugged into any convenient AC outlet near the computer. On command from the computer, the command console sends signals to remote modules located at the devices to be controlled. Up to 16 remote modules may be controlled.

Introl/X-10 comes with software to control devices on predetermined schedules. It provides control of devices at a given time, selection of daily or weekly schedules, specification of day of the week or exact date for a particular event, specification of intervals of time for an event and device wattage ratings for power consumption accounting for energy management. Used with Mountain Hardware's Apple Clock, the system provides "foreground"/"background" capability.

The system, consisting of the Introl Controller board with timer and ultrasonic transducer, the BSR/X-10 Command Console and three remote modules is priced at $279. The Introl/ X-10 Controller Card separately sells for $189. Additional remote modules are available at $15. For more information, contact Mountain Hardware, Inc., 300 Harvey West Blvd., Santa Cruz, CA 95060. *Circle No. 127*

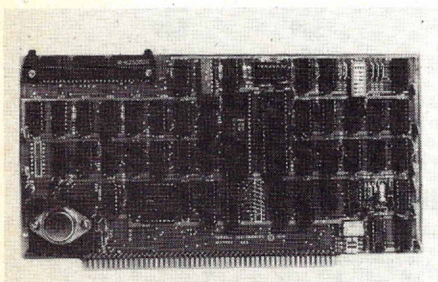
Double Density Floppy Interface

Tarbell Electronics has released a double density floppy disk interface, a component which enhances existing disk storage capacities with minimum reconfiguration of existing microcomputer systems, according to the company.

The interface board is supplied with the new BASIC Input/Output System software for CP/M on single-density diskette, permitting the user to intermix single and double density diskettes. The Tarbell system automatically determines whether single or double density is in use. As many as four drives can be selected, using either single or double density.

Additional features provide a combination of versatility, speed and reliability. For instance, the 8″ Shugart-compatible disk interface contains

phase-lock-loop and write precompensation, providing reliable data storage and recovery, the company added. In addition, the on-board phantom bootstrap PROM is disabled on completion



of the bootstrap operation, freeing all 64K of memory address space for other use. Multi-user operation is now possible due to the direct memory access for either single or double-density operation. Extended addressing capability provides 8 additional address bits as specified by the new IEEE standard, allowing direct transfers to and from any location within a 16 megabyte address range.

The interface is shipped assembled and factory tested, with BIOS for CP/M on diskette, and a 6-month warranty on parts and labor. Priced at $425, the interface is available for immediate delivery. For more information, contact Tarbell Electronics, 950 Dovlen Place, Suite B., Carson, CA 90746; (213) 538-4251. *Circle No. 142*

RAM Board for Heath H8

D-G Electronic Developments Co. introduced their new random access memory for Heath H8 computers. The RAM board, configured with 32K bytes consumes less than 6 watts power.

Fully assembled, tested and burned-in, the DG-32D is ready to plug into the H8 and use without additional wiring. The DG-32D is designed to operate either with or without the present static memory in the computer.

Features include full compatibility with current Heath peripherals; circuit protection to prevent damage to memory output buffers if two blocks are assigned to the same address space; memory addressing controlled by DIP switch; 4 independently addressable 8K blocks; transparent refresh; no wait states required.

Price for the board is $479. Visa and Master Charge are accepted. The DG-32D is fully guaranteed for one year.

For more information contact Pacesetters Graphics, 3223 Forest Lane, Garland, TX 75042. *Circle No. 149*

LITERATURE

Educational Comic Book

A new, updated version of Radio Shack's educational comic book, "The Science Fair Story of Electronics . . . the Discovery That Changed the World!", is now available for free distribution to schools, clubs, youth groups and interested individuals.

The booklet, designed as a motivational learning aid for young people, focuses on the human interest side of science with important discoveries and the people who made them described in easy-to-read narrative.

Topics include magnetism, ancient use of batteries, electricity in nature, the development of "wireless" communications, TV, radar and the transistor, electronics in aviation and space exploration, and the computer age.

The booklet is available from participating Radio Shack stores and dealers.

For more information contact Radio Shack, 1300 One Tandy Center, Fort Worth, TX 76102. *Circle No. 135*

Student Workbooks for Pet

Cow Bay Computing has published two student workbooks for beginning programming classes using the 8K Pet.

Both the beginner and intermediate workbooks have been designed for schools which are inserting a computer literacy unit into regular math or science classes, the company said. Non-computer oriented teachers, as well as those with prior skills, can use the workbooks as complete lesson plans with assignments included, the company added.

Teaching units in the volumes feature step-by-step instruction, daily lesson plans, classwork and homework worksheets, quizzes, hands-on and

CLASSIFIEDS

Radio Shack® DEALER

COMPUTER CENTER

MICRO MANAGEMENT SYSTEMS

Up To 15% Discount on TRS-80's

WE HAVE THE HIGHLY RELIABLE LOBO DISK DRIVE IN STOCK!

MINI MALL—DOWNTOWN SHOPPING CENTER
CAIRO, GEORGIA 31728
912-377-7120

TRS-80 Quality Software

	32K Mod I	64K Mod II
Data base manager	$69	$199
Mailing list system	$59	$99
Inventory system	$99	$149
Accounts receivable	$69	$149
Word processor	$49	$49
Manual only	$5	$10

Prices include disk, documentations, postage.

Tony W. Pow
96 Dothan St.
Arlington, MA 02174

The Best of **Personal Computing** For Your Home and Business

See page 21.

Looking to buy or sell your software? Try Personal Computing Classifieds.

Rates:	**Per col. inch**
Up to 29 column inches	**$40.00**
30-59 column inches	**$36.00**
60-99 column inches	**$33.00**
100-299 column inches	**$30.00**
300-599 column inches	**$27.00**
More than 599 column inches	**$24.00**

Call **Personal Computing** (617) 232-5470

Moving?

1. For FASTEST service attach old mailing label in space below.

If mailing label is not available print your old name and address in this box

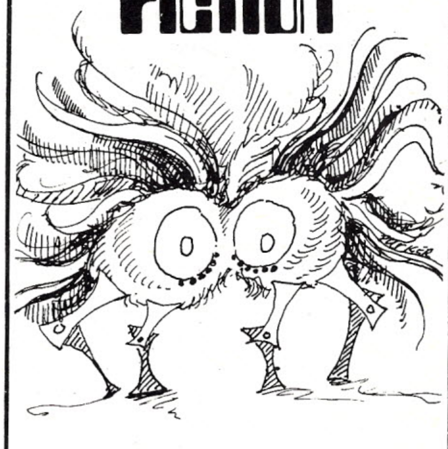
Please allow 6 weeks for change to take effect

2. Print your NEW address here

NAME ______
TITLE ______
COMPANY ______
ADDRESS ______
CITY ______ STATE ______ ZIP ______

3. Mail to: Circulation Manager
Personal Computing
1050 Commonwealth Ave.
Boston, MA 02215

Science Fiction



Turn your daydreams and visions of the future into cash. We're looking for short (500 to 2000 words) science fiction stories dealing with the future of microcomputers — their possible uses and their roles in society and in people's lives. Stories must be original and not published elsewhere. Submit your typed, double-spaced story to Personal Computing, 1050 Commonwealth Ave., Boston, MA 02215.

---

save more than 20%

NORTH STAR — INTERTUBE
THINKER TOYS — MICROTEK

The smartest computers at the smartest price

QUAD & DOUBLE DENSITY

	LIST	ONLY
HORIZON-1-32K-D kit	$1999	**$1585**
HORIZON-2-32K-D kit	2399	**1905**
Assembled & tested	2765	**2195**
HORIZON-2-32K kit QUAD	2799	**2225**
Assembled & tested	3215	**2555**
PASCAL for NORTH STAR on Disk		**49**
Powerful NORTH STAR BASIC		**FREE**
THINKER TOYS DISCUS/ 2D A&T	1149	**949**
INTERTUBE II SMART TERMINAL	995	**745**
MARYELLEN WORD PROCESSOR Your Best Buy		**38**
MICROTEK PRINTER	750	**675**
PDS $99 Anadex Printer	995	**875**
COMPILER for HORIZON Secret Superfast Code		**100**
VERBATIM for the best DISKETTES	Box of 10	**29**
Which Computers are best?	BROCHURE. . .	**FREE**

AMERICAN SQUARE COMPUTERS
KIVETT DR * JAMESTOWN NC 27282
(919) 883-1105

hands-off exercises and group and individual assignments.

Prices for the volumes are : "Feed Me, I'm Your Pet" (Beginner-1), $4.95; "Looking Good with your Pet" (Intermediate-1), $4.95; and "Teacher's Pet" (Lesson plans, quizzes and answer key), $4. "Teacher's Pet" is free with orders of 25 or more workbooks. Quantity discounts are available. For more information contact Cow Bay Computing, Box 515, Manhassett, NY 11030. *Circle No. 137*

Electronics Catalog

A 104-page catalog of electronic kits is now available free from Heath Company. New products featured include a series of self-instruction math programs and an educational program entitled "Computer Concepts for Small Business"

Product categories include personal computer systems, amateur radio, color television, stereo components, test instruments, digital clocks, auto, marine and aircraft accessories, plus a new selection of self-instruction programs in electronics-related subjects.

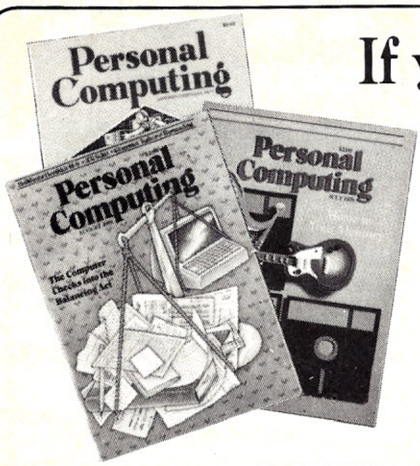
For more information contact Heath Company, Dept. 350-930, Benton Harbor, MI 49022. *Circle No. 144*

1980 Radio Shack Catalog

Radio Shack's 176-page 1980 catalog is now available free from more than 6000 participating stores and dealers nationwide. The catalog features computers and stereo components, toys, parts and accessories for home entertainment, hobbyists and experimenters.

The catalog includes the TRS-80 system, the new Model II business microcomputer and the complete line of Realistic stereo components, CB equipment, radios, tape recorders, Archer antennas, Micronta test instruments and ArcherKit and Science Fair hobby kits.

For more information contact Radio Shack, 1300 One Tandy Center, Fort Worth, TX 76102; (817) 390-3272. *Circle No. 148*



If you're missing any of these you have gaps in your data bank.

Update your files with **Personal Computing** back issues. Past issues contain articles on computer basics and getting acquainted with your personal system, programs for business and home use, educational applications, programming tips and suggestions on getting the most from your computer – plus much, much more.

Just send $3 for each issue* of **Personal Computing**. Use the coupon below to place your order while our limited supply lasts.

* Canada and Mexico, $4; all other foreign, $6.

□ Send me these back issues of **Personal Computing** for $3 each:

- □ January/February 1977
- □ July/August 1977
- □ September/October 1977
- □ November/December 1977
- □ January 1978
- □ April 1978
- □ May 1978
- □ June 1978
- □ October 1978
- □ November 1978
- □ June 1979
- □ July 1979
- □ October 1979
- □ November 1979
- □ December 1979

Issues not listed are out of print

□ My check is enclosed.
□ Charge to my □ Master Charge □ Visa
□ Account No. ______________
Card Expiration Date ______________

Name ______________________________

Address ______________________________

City ____________ State ______________ ZIP ________

Mail to: **Personal Computing**
1050 Commonwealth Ave.
Boston, MA 02215

ADVERTISERS' INDEX

SALES OFFICES

Northeast: Arlene Joseph, Benwill Publishing Corp., 2 Park Ave., New York, NY 10016; (212) 340-9780 • **Midwest:** Pat Bosecke, P.O. Box 56089, Harwood Heights, IL 60656; (312) 631-7319 • **Northern California:** Ralph Peterson, 1380 Oak Creek Dr., #212, Palo Alto, CA 94034; (415) 328-4392 • **California:** Joanna Spiro, Mike Reynolds, 924 Westwood Blvd., Los Angeles, CA 90024; (213) 478-3017 • **Japan:** Hiro H. Irie, International Business Corp., 11-8, Narita-Higashi 1-chome, Suginami-ku, Tokyo 166; (03) 311-1746

Step up to your next computer.

STEP UP TO A C4P FROM OHIO SCIENTIFIC

You know about computers. In fact, you probably own one now. One that you might be thinking of expanding. We have a better idea. Take a giant step into the personal computing future with an amazing, new C4P from Ohio Scientific.

SPEED SEPARATES THE COMPUTERS FROM THE TOYS

The C4P MF has execution speed that is twice as fast as Apple II or Commodore PET and over THREE times as fast as TRS-80. They are many times faster than the recently introduced flock of video game type computers. And, as if that weren't fast enough, the C4P nearly doubles its speed when equipped with the GT option.

Just look at the back panel of the C4P MF.

All the I/O you'll ever need!

SOUND

1—programmable tone generator 200 — 20KHz
1—8 bit companding digital to analog converter for music and voice output.

HUMAN INPUT EXPANSION

2—8 axis joystick interfaces
2—10 key pad interfaces

HOME INTERFACE

1—AC-12 AC remote control interface

DISPLAY

32 x 64 with upper and lower case
2048 Characters.
256 x 512 effective Graphic Points
16 Colors

SOFTWARE

Ohio Scientific offers a comprehensive library of both systems and applications software for the C4P.

The C4P is an outstanding premium computer — years ahead of the market. We know because there's nothing quite like it for the price, anywhere. And probably won't be for a very long time.

C4P $698

8K BASIC-in-ROM, 8K of static RAM and audio cassette interface. Can be directly expanded to 32K static RAM and two mini-floppy disks.

C4P MF $1695

All the features of the C4P plus real time clock, home security system interface, modem interface, printer interface, 16 parallel lines and an accessory BUS. The C4P MF starts with 24K RAM and a single mini-floppy and can be directly expanded to 48K and two mini-floppies. Over 45 diskettes now available including games, personal, business, educational and home control applications programs as well as a real time operating system, word processor and a data base management system.

Computers come with keyboards and floppies where specified. Other equipment shown is optional.

For literature and the name of your local dealer, CALL 1-800-321-6850 TOLL FREE.

OHIO SCIENTIFIC

1333 SOUTH CHILLICOTHE ROAD
AURORA, OH 44202 • (216) 562-3101